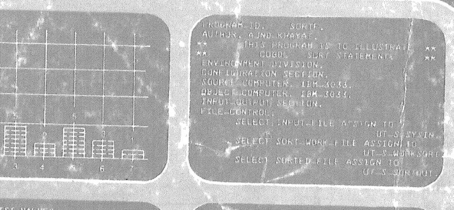


إدارة البحوث

البرمجة التطبيقية بلغة كوبول



LIST VALUES

QUANTITY	PRICE	AMOUNT
20.00	20.00	400.00
34.00	12.00	408.00
67.00	23.00	1541.00
50.00	34.00	1700.00
35.50	23.30	827.15
25.50	45.50	1160.25
TOTAL		4036.40



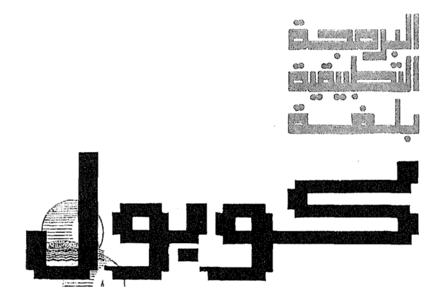
أحمد جنيد جن...

عضو هيئة التد...

١٤٠٣هـ



إدارة البحوث



محمد بسام زهير الخياط
عضو هيئة التدريس بالمعهد

أحمد جنيد جنيد
عضو هيئة التدريس بالمعهد

١٤٠٣هـ

المحتويات

الصفحة

الصفحة

الصفحة

الصفحة

الصفحة

الصفحة

الصفحة

الصفحة

تقـــــديم

تـشـكـل الـتـطـبـيقات الإدارية والتجارية على الحاسبات الالكترونية نسبة كبيرة بـالمقـارنة مع التطبيقات الأخرى، ولذلك كان للغة البرمجة كوبول دور واسع وكبير في كـتابة وتطوير البرامج والأنظمة ذات الطبيعة الإدارية والتجارية لأنها مصممة ومطورة لتفي بهذا الغرض بشكل رئيسي.

وانـطـلاقـاً من هذا الاستخدام الواسع لهذه اللغة العالية من لغات البرمجة وتمشياً مع الـتـطـويـر الـسـريـع والـنـهضة العلمية في المملكة فقد قمنا باعداد هذا الكتاب ليكون مساعداً لقراء اللغة العربية في فهم لغة البرمجة هذه والتعرف على كيفية استخدامها.

ولا تـنـحصر الاسـتفادة من هذا المؤلف بالدارسين في دبلوم الحاسب الآلي في المعهد بل يمكن أن يكون مرجعاً لهذه اللغة والتدرب عليها في مختلف المستويات.

وقد اعتمدنا عند وضع هذا المؤلف في جميع أقسامه وبفصوله على لغة كوبول المطورة عـام ١٩٧٤ والـتـي وافـقـت عليها هيئة ANSI للغات. علاوة على الأخذ بعين الاعتبار المـواضـيع التي طورت بعد حلول الحاسبات الصغيرة Mini Computer والتي تعتمد على الاسـلـوب المباشر في التشغيل، مما يجعل المؤلف مفيداً في التدريب على كلا النوعين من الحاسبات الصغيرة والكبيرة وطريقتي التشغيل المباشرة وغير المباشرة.

وقد روعي عند وضع هذا المؤلف الغرض التدريبي والتعليمي منه وذلك باللجوء الى تسلسل خاص للمواضيع المطروقة . والذي قد يختلف مع المؤلفات المماثلة ، بحيث يمكن للدارس استخدام الحاسب والتدرب عليه وفق هذه اللغة بأقصر مدة ممكنة . ولذلك فقد تم التركيز على قسم الاجراءات في برنامج كوبول واعطاؤه الأولية في الشرح والبحث لأنه جوهر البرنامج وأساسه المنطقي .

وقد تم تقسيم هذا المؤلف إلى أربعة أقسام رئيسية :

القسم الأول : و يتكون من ثلاثة فصول تبحث في أساسيات لغة كوبول وأقسام البرنامج الأساسية وكيفية كتابة برنامج بسيط بهذه اللغة .

القسم الثاني : و يتكون من ستة فصول تبحث في التعليمات والعبارات المتقدمة في لغة كوبول مما يعطي المبرمج المبتدىء المعلومات اللازمة لتحسين وتطوير برامجه البسيطة .

القسم الثالث : و يتكون من فصلين ويحتوي على المواضيع الخاصة باستخدام الحاسب في كتابة وترجمة وتنفيذ برنامج كوبول .

القسم الرابع : و يتكون من فصل واحد ويحتوي على تطبيقات عملية شاملة لمختلف المواضيع المهمة والأساسية في هذه اللغة مثل انشاء الملفات وتعديلها وطباعتها .

كما اشتمل المؤلف على ملحقات خاصة بالموضوعات الضرورية لتساعد المبرمج في الرجوع اليها عند كتابة برامج كوبول .

ونتوجه بشكرنا وتقديرنا الى إدارة المعهد للتشجيع والمساعدة الذين لاقيناهما عند إعداد هذا المؤلف وإبرازه الى حيز الوجود .

كما اننا نشكر قسم الطباعة والنشر وكافة العاملين فيه على الجهد الكبير اللازم لاظهار هذا المؤلف بالشكل اللائق والمطلوب.

ونأمل أن نكون قد وفقنا في وضع هذا المؤلف ليكون عوناً للدارسين والمهتمين والطلاب في فهم واستخدام لغة كوبول وليكون لبنة من اللبنات الأولى في بناء صرح المكتبة العربية المعاصرة.

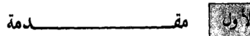
والله من وراء القصد

٢٢ رمضان ١٤٠٣هـ

٣ تموز ١٩٨٣م

المؤلفان





الفصل الأول — مقـــــــــــــــدمة



التعليمات الأولية في لغة الكوبول (قسم الإجراءات)



الأقســـام الأخـرى في برنامـج كوبـــول



الفصل الأول

مقـــــــــــدمة

١ ـ ١ لمحة عامة*

تعتبر لغة الكوبول (COBOL) المختصرة من الكلمات الانكليزية التالية COmmon Business Oriented Language (وتعني اللغة الموجهة لإدارة الأعمال العامة) من أوسع لغات البرمجة العالية انتشاراً. وقد صنفت مع اللغات العالية بسبب سهولة قراءتها وقُربها الى اللغة الانكليزية العادية، ويمكن كتابة البرامج بها بعيداً عن الرموز والاصطلاحات الغامضة.

وقد بدأ العمل في تطوير لغة الكوبول تاريخيا منذ عام ١٩٥٩ ميلادي (مايو) في الولايات المتحدة من قبل المستفيدين في حقل إدارة الأعمال من الحاسب الآلي. وصدرت النسخة الأولى (Version) في نفس العام في شهر ديسمبر ثم تلتها النسخة المنقحة عام ١٩٦١ ميلادي وسميت COBOL - 61 والتي تعتبر حجر الأساس للتحسينات والتطوير الذي طرأ على النسخ التالية. وفي عام ١٩٦٨ ميلادي تمت الموافقة على النسخة المعروفة باسم ANSI COBOL من قبل هيئة تسمى American National Standard Institute.

وبعد ذلك استمرت التحسينات والإضافات على لغة الكوبول وكذلك حذفت الأشياء الغامضة وغير الواضحة بحيث أقرت هيئة ANSI النسخة الجديدة والأخيرة عام ١٩٧٤ وسميت ANSI COBOL 74.

(*) Structured COBOL, Philippakis and Kazmier, MacGraw Hill, 1981 P. 17 , 18.

وتعتبر الميزة الأساسية والأولى للغة كوبول أنها لغة موثقة ذاتياً أي Self-documente لأن المرونة المتوفرة في اختيار أسماء الحقول والمتغيرات كبيرة وكذلك لكونها قريبة جداً من اللغة الانكليزية. وهذه الميزة تسمح لمراجع البرنامج فهمه ومعرفة وظائفه بسرعة وكذلك مهمة جداً لأجل صيانة وتطوير البرنامج في المستقبل مما يجنب المبرمج إعادة كتابة نفس البرنامج عند عدم وضوحه ومعرفة وظائفه من أجل عملية التطوير والتحسين على البرنامج التي تعتبر أساسية في عملية البرمجة. ولكن الى جانب إمكانية كتابة برنامج واضح وسهل ومقروء بلغة كوبول يمكن كتابة برامج غامضة عند استخدام أسماء غير واضحة ورموز بدلا من الأسماء المعبرة عن المتغيرات والحقول المستخدمة في البرنامج.

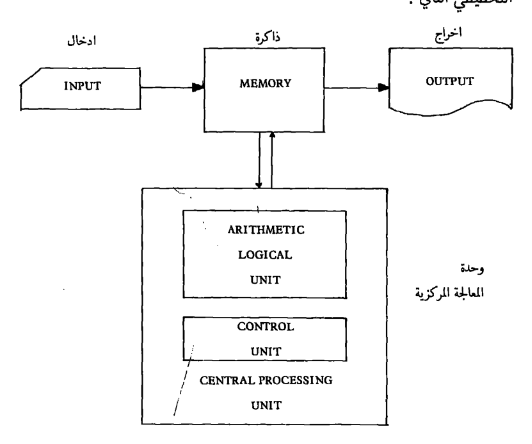
وعلى الرغم من كون لغة كوبول مستخدمة منذ زمن بعيد نسبياً إلا أن استخدامها قد زاد في فترة السبعينات (١٩٧٠م وما بعد). وانتشارها الواسع بعد ذلك التاريخ أدى الى ظهور نسخ (Versions) محسنة من هذه اللغة تتمشى مع ظهور الحاسبات الصغيرة Mini Computer وحلول الشاشة المتلفزة «VDU» لاستقبال وإظهار البيانات. ولكن تبقى الاقسام الأساسية في لغة كوبول ثابتة، وهذه التحسينات تعتبر إضافات ثانوية إلى حين اعتمادها من قبل الهيئة المسماة ANSI.

وتجدر الاشارة أخيراً إلى أن عملية كتابة البرامج بلغة كوبول كانت غير مستحبة في السابق لأنها تحتاج الى وقت أطول اثناء التنفيذ على الحاسب ولكن بعد تطوير المترجمات COBOL COMPILERS أصبحت سرعة تنفيذ البرامج المكتوبة بهذه اللغة قريبة الى نفس السرعة التي تحتاجها البرامج المكتوبة بلغة ذات مستوى أدنى Low-Level. ولذلك تعتبر لغة كوبول اليوم من اللغات الأساسية في معالجة البيانات الإدارية والأكثر شيوعاً في الاستخدام وسوف تبقى كذلك لحين ظهور لغة برمجة لها نفس الصفات وبفعالية أكبر.

١ ـ ٢ الحاسب الآلي وبرنامج كوبول :

من المعروف أن الحاسب الآلي عبارة عن مجموعة من الأجهزة الالكترونية التي تقبل المعلومات وتعالجها وترصد نتائجها وفق خطة محددة تسمى البرنامج.

و يتألف الحاسب الآلي من مجموعة من الأجهزة أو الوحدات تسمى وحدات إدخال أو وحدات إخراج ومن وحدة المعالجة المركزية كما هومبين في الشكل التخطيطي التالي :



والبرنامج عبارة عن مجموعة من التعليمات المتسلسلة تسلسلاً منطقياً لأداء غرض معين، فهو يقرأ البيانات من وحدات إدخال مثلاً و يعالج هذه البيانات ثم يظهر نتائجها على وحدات إخراج وهكذا ..

ويمكن للبرنامج أن يصاغ بلغات حاسب مختلفة، هذه اللغات تنقسم الى قسمين :

- لغات دنيا LOW-LEVEL LANGUAGES مثل اسمبلر ASSMBLER.
- لغات عليا HIGH-LEVEL LANGUAGES مثل كوبول، فورتران، بيزك...

إن لكل لغة مزايا خاصة، و يتحكم في اختيار لغة البرنامج عدة اعتبارات أهمها طبيعة النظام أو البرنامج. وتعتبر لغة كوبول من لغات البرمجة العليا، والبرامج المكتوبة بهذه اللغة تعتبر بصيغة المصدر أي SOURCE وتحول الى لغة دنيا خاصة بالحاسب عبر إمرارها على برنامج مترجم يسمى - المترجم COMPILER - الذي يحولها إلى صيغة التنفيذ OBJECT*

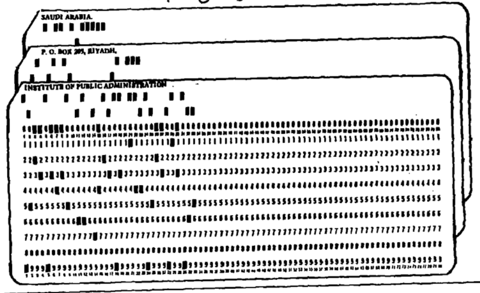
ولإعطاء فكرة سريعة و واضحة من برنامج كوبول نأخذ المثال التالي :

لدينا ملف من البطاقات المثقبة يحتوي على اسم معهد الادارة العامة وعنوانه :

INSTITUTE OF PUBLIC ADMINISTRATION

P.O. BOX 205, RIYADH, SAUDI ARABIA

المطلوب قراءة هذه البطاقات وطباعة محتواها على الطابعة.



(*) تمثل لغة الآلة القابلة للتنفيذ بعد الترجمة

ولا بد قبل وضع أي برنامج من رسم مخططه الانسيابي أو المنطقي flowchart (*)

وقبـل البدء في رسم المخطط الانسيابي للمسألة، نتعرف على الرموز المستخدمة في رسم المخطط الانسيابي وهي :

رمز البداية أو النهاية	START / END	
رمز المعالجة	PROCESS	
رمز القراءة أو الكتابة	READ / WRITE	
رمز اتخاذ قرار	DECISION	
صلة وصل	CONNECTOR	
اتجاه التدفق أو السير	DIRECTION OF FLOW	⟶

الرموز المستخدمة حسب

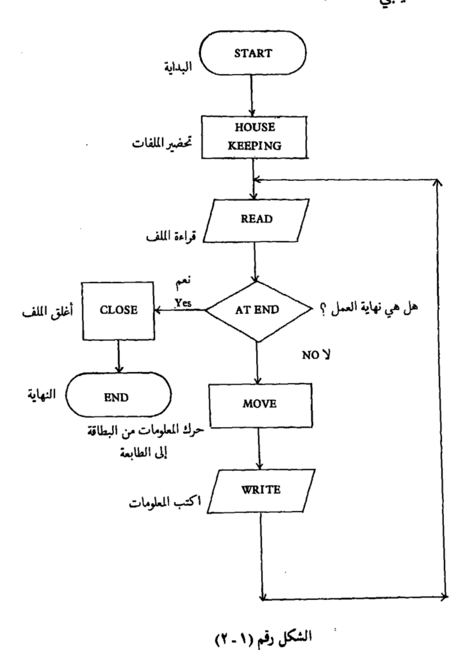
AMERICAN NATIONAL STANDARDS INSTITUTE FLOWCHART SYMBOLS

الشكل رقم (١ - ١)

ومن استخدام هذه الرموز يمكن وضع أي مخطط انسيابي لبرنامج حاسب آلي فيكون المخطط المطلوب هو التالي والذي يوضح سير البرنامج منطقياً.

(*) المعجم العربي الموحد لمصطلحات الحاسبات الالكترونية ـ المنظمة العربية للعلوم الإدارية بجامعة الدول العربية ـ عمان ١٩٨١، ص ١٢٨.

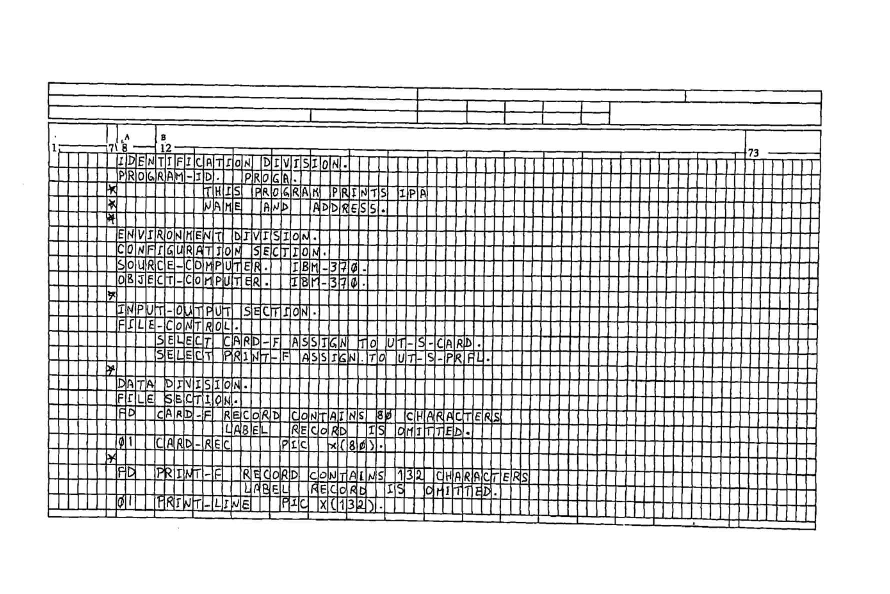
المخطط الانسيابي للمسألة :



الشكل رقم (١ - ٢)

وفي الصفحة التالية ترميز لهذا المخطط الانسيابي وفق قواعد لغة كوبول .

```
1      7 8   12                                                       73
      *WORKING-STORAGE SECTION.
      *
        77  END-OF-JOB   PIC X(3) VALUE 'NO'.
      *
       PROCEDURE DIVISION.
       STARTT.
           OPEN  INPUT  CARD-F
                 OUTPUT PRINT-F.
      *
           PERFORM  READ-WRITE THRU EXT
                    UNTIL END-OF-JOB = 'YES'.
           CLOSE CARD-F   PRINT-F.
           STOP RUN.
      *
       READ-WRITE
           READ  CARD-F AT END MOVE 'YES'
                 TO END-OF-JOB  GO TO EXT.
      *
           MOVE  CARD-REC  TO PRINT-LINE.
           WRITE PRINT-LINE  AFTER 1 LINE.
      *
       EXT.
           EXIT.
```



```
1     7 8   12                                                      73
       IDENTIFICATION DIVISION.
       PROGRAM-ID.  PROGA.
      *              THIS PROGRAM PRINTS IPA
      *              NAME  AND  ADDRESS.
      *
       ENVIRONMENT DIVISION.
       CONFIGURATION SECTION.
       SOURCE-COMPUTER.  IBM-370.
       OBJECT-COMPUTER.  IBM-370.
      *
       INPUT-OUTPUT SECTION.
       FILE-CONTROL.
           SELECT CARD-F ASSIGN TO UT-S-CARD.
           SELECT PRINT-F ASSIGN TO UT-S-PRFL.
      *
       DATA DIVISION.
       FILE SECTION.
       FD  CARD-F RECORD CONTAINS 80 CHARACTERS
                   LABEL  RECORD  IS OMITTED.
       01  CARD-REC       PIC  X(80).
      *
       FD  PRINT-F  RECORD CONTAINS 132 CHARACTERS
                     LABEL  RECORD  IS  OMITTED.
       01  PRINT-LINE   PIC X(132).
```

و يلاحظ من هذا المثال أن البرنامج يتألف من أربعة أقسام رئيسية وهي :

١ ـ قسم التعريف IDENTIFICATION DIVISION

و يستخدم هذا القسم لتعريف البرنامج باسم محدد وكذلك اسم كاتب البرنامج و يبدأ المبرمج كتابة هذه الفقرات من القسم A من العمود ٨ على صفحة الترميز الخاصة بلغة كوبول.

٢ ـ قسم البيئة (المحيط) ENVIRONMENT DIVISION

وهذا القسم يحتوي على توصيف الحاسب المستخدم وكذلك وحدات الإدخال والإخراج وذلك من أجل ربط الأسماء المذكورة في البرنامج بهذه الوحدات (الأسماء المنطقية) مع الأسماء الفعلية المعرفة في نظام التشغيل. و يقوم هذا القسم في البرنامج بدور الوسيط بين البرنامج نفسه ونظام التشغيل. لذلك لا بد من الرجوع الى المراجع المتوفرة عن الحاسب المستخدم أثناء كتابة هذا القسم.

قسم البيانات DATA DIVISION

يحتوي هذا القسم توصيف وتحديد البيانات المستخدمة في البرنامج ويمكن أن يشتمل على ما يلي :

- وصف بيانات الادخال والاخراج من / إلى البرنامج ضمن الفصل File Section وكلمة FD تعني File Description و يلاحظ أنها تبدأ من العمود ٨ ـ ١١ و بعد تعريف الملف يرد ذكر لوصف سجل من هذا الملف المستخدم برقم المستوى 01.
- وصف للحقول أو الأماكن المحجوزة في الذاكرة لغرض الاستعمال الداخلي في البرنامج فصل WORKING-STORAGE ويحتوي في هذا المثال على الحقل END - OF - JOB و يأخذ رقم المستوى 77.

و يـلاحـظ أن أرقام المستوى وأسماء الفصول في هذا القسم تبدأ من العمود ٨ ـ ١١ في صفحة ترميز كوبول .

٤ ـ قسم الاجراءات PROCEDURE DIVISION

يحتوي هـذا الـقـسم على جوهـر الـبـرنامج والتعليمات أو العبارات ذات الصيغة الأمـريـة للحـاسـب وتنفذ هذه الاجراءات حسب تسلسل ورودها و يلاحظ استعمال تعليمة OPEN لغرض فتح الملف مع تحديد نوع الملف INPUT أو OUTPUT ثم تعليمة PERFORM الـتـي تـطـلـب إنجاز الفقرة READ-WRITE عددا من المرات لغاية تحقق الـشـرط 'END - OF - JOB = 'YES ثـم تـرد تـعـلـيـمـة إغلاق الملف بعد الانتهاء منه CLOSE وتـعـلـيـمـة نـهاية البرنامج المنطقية STOP RUN . أما الفقرة المنفذة عددا من المـرات فـهـي تشتمل على عملية قراءة البطاقات المثقبة READ وفي نهاية البطاقات يتم نـقـل القيمة الحرفية الثابتة YES إلى الحقل END - OF - JOB ثم نقل محتوى البطاقة إلى سـجـل (أو سـطـر) الـطـبـاعة ثم تعليمة الطباعة أو الكتابة WRITE على الطابعة (ملف الطباعة) بعد التقدم سطر واحد AFTER 1 LINE ثم تعليمة نهاية الفقرة EXIT .

و يـلاحـظ في قـسم الاجراءات أن أسماء الفقرات تـبدأ ضمن الأعمدة ٨ ـ ١١ أما التعليمات ومحتوى الفقرة نفسها فيرد بعد العمود ١٢ ولغاية ٧٢ .

وسوف يتم ذكر تفاصيل أكثر عن هذه الأقسام في الفقرات والفصول التالية .

أمـا إشـارة الـنجمة (*) في العمود ٧ فهي تعني أن السطر المذكورة فيه يعتبر كسطر ملاحظات ضمن البرنامج ولا يتم تنفيذه .

١ ـ ٣ الخطوات المنطقية لكتابة برنامج كو بول

يمـر بـرنـامـج كـو بـول بـعـدة مـراحـل قبل أن يصبح جاهزاً للتشغيل ، هذه المراحل مـتـسـلـسـلـة تـسـلـسـلاً مـنـطقياً وعملياً ولا يمكن تقديم أو تأخير خطوة عن خطوة وهذه

الخطوات هي :

١ -	تعيين وتحليل المسألة	PROBLEM STATEMENT AND ANALYSIS
٢ -	رسم مخطط انسيابي للعمل	FLOWCHARTING
٣ -	ترميز البرنامج	CODING
٤ -	تمرير البرنامج على المترجم	COMPILER
٥ -	تنفيذ البرنامج بعد الترجمة	EXECUTION
٦ -	توثيق البرنامج	DOCUMENTATION

١ - تعيين وتحليل المسألة PROBLEM STATEMENT AND ANALYSIS

يجب علينا أولاً تفهم المسألة المعطاة، والمدخلات لهذه المسألة وشكل المخرجات المطلوب، وكذلك طريقة المعالجة بالتفصيل، وكلما كان فهم المسألة، أعمق كلما سهل علينا رسم مخطط انسيابي صحيح لها ومتابعة حل البرنامج بصورة سليمة وسريعة.

٢ - رسم مخطط انسيابي للمسألة (البرنامج) PROGRAM FLOWCHART

بعد تفهم المسألة (البرنامج) جيداً يمكن أن نبدأ برسم المخطط الانسيابي. وهو عبارة عن أشكال ترمز إلى عمليات معينة - انظر الشكل رقم (١ - ١) - ويمكننا ترتيب هذه الرموز بشكل منطقي حسب معطيات المسألة وحسب الحل الذي نراه مناسباً وكلما كان المخطط الانسيابي جيداً ودقيقاً كلما استطعنا أن نحول هذه الأشكال الى تعليمات خاصة بلغة كوبول بشكلٍ سليم. انظر الشكل (١ - ٢).

٣ - الترميز CODING

وهذه المرحلة تتضمن تحويل المخطط الانسيابي الى لغة كوبول و يقوم بها المبرمج طبعاً، إذ يحول أو يترجم كل خطوة من خطوات هذا المخطط الى تعليمة كوبول أو

أكثر، ويجب أن تتم عملية الترميز بدقة فائقة و باتباع أصول وقواعد لغة كوبول ، وتتم عملية الترميز على ورق خاص يسمى COBOL CODING SHEETS وعند الانتهاء من كتابة البرنامج على هذا الورق يكون البرنامج جاهزاً و بصيغة تسمى صيغة الأصل SOURCE . انظر الشكل رقم (١ - ٣) .

٤ ـ عملية الترجمة COMPILATION

بعد عملية الترميز يُدخل البرنامج للحاسب عن طريق وحدات ادخال مختلفة ، اما عن طريق بطاقات مثقبة ومن ثم قارىء البطاقات CARD READER أو عن طريق شريط ممغنط KEY TO TAPE والشكل الشائع حالياً والأسهل استعمالاً عن طريق النهائيات TERMINALS باستخدام برنامج منقح النصوص TEXT EDITOR و يتم التخزين إما على قرص ممغنط* DISC أو على قرص صغير يسمى DISKETTE .

بعد عملية التخزين هذه يتم تمرير هذا البرنامج على برنامج موجود في الحاسب يدعى البرنامج المترجم COMPILER و يقوم المترجم هذا بتحويل برنامج كوبول من صيغة الأصل SOURCE الذي كنا قد كتبناه في الخطوة رقم ٣ ـ الترميز ـ الى صيغة خاصة بالحاسب MACHINE LANGUAGE . و يكون البرنامج عندها بصيغة قابلة للتنفيذ ـ عند خلوه من الأخطاء القاعدية طبعاً ـ وتسمى هذه الصيغة OBJECT .
ناتج عملية الترجمة COMPILATION هذه يمكن أن يحتوي على أخطاء قاعدية SYNTAX ERRORS . فيجب أن تصحح هذه الأخطاء وتعاد عملية الترجمة COMPILATION من جديد، ولا نستطيع الانتقال للخطوة التالية حتى يكون البرنامج خالياً من الأخطاء FREE OF ERRORS .

(*) تكتب أحياناً DISK

٥ ـ تنفيذ البرنامج EXECUTION

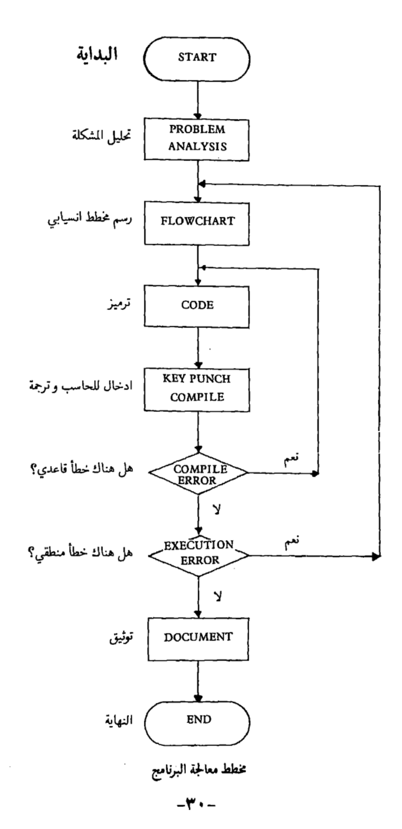
بعد أن يكون البرنامج قد نقح تماماً من الأخطاء القاعدية وتشكل له OBJECT أي صيغة التنفيذ، نستطيع تجربة البرنامج أي عمل TEST أو RUN له. بعد تحضير الملفات الخاصة ان وجدت. بعد عملية التنفيذ هذه ندقق النتائج OUTPUTS، فإذا احتوت على أخطاء نرجع ونعدل أصل البرنامج SOURCE وكذلك المخطط الانسيابي لهذا البرنامج إن لزم الأمر، هذه الأخطاء تسمى الأخطاء المنطقية LOGICAL ERRORS، ثم تعاد عملية الترجمة COMPILATION من جديد، وهكذا حتى نحصل على نتائج سليمة تماماً. عندها فقط يكون البرنامج صحيحاً وجاهزاً للاستعمال.

٦ ـ التوثيق DOCUMENTATION

إن مرحلة التوثيق تعتبر مرحلة مهمة على المدى البعيد، إذ أنه بعد فترة من الزمن يمكن أن تتغير معطيات المسألة ـ البرنامج ـ ونحتاج الى تعديل البرنامج من جديد، ومن الممكن جداً أن يكون المبرمج الذي كتب هذا البرنامج قد نقل الى مكان آخر أو قد بدل عمله أو أي احتمال آخر، لذا لا يشترط بعملية صيانة البرنامج ـ تعديل البرنامج ـ أن يقوم بها نفس المبرمج الذي كتبه، لذلك يجب على كل مبرمج بعد الانتهاء من كتابة برنامجه تماماً والتأكد من خلوه من الأخطاء القاعدية والمنطقية أن يحتفظ بكل المعلومات اللازمة عن هذا البرنامج وأن يحتفظ بآخر نسخة صحيحة عن البرنامج ونسخة من المدخلات INPUTS ونسخة من المخرجات OUTPUTS، والمخطط الانسيابي للبرنامج PROGRAM FLOWCHART. يحتفظ بذلك في ملف خاص يسلم لمدير المشروع أو للمسئول عن النظام.

بذلك يكون المبرمج قد كتب برنامجاً متكاملاً بأسلوب صحيح و بالسرعة والدقة المطلوبة.

الشكل التالي يوضح تسلسل هذه العمليات :



مخطط معالجة البرنامج

١ ـ ٤ العناصر المكونة للغة كوبول

لغة كوبول هي لغة تخاطب بين الانسان والحاسب الآلي مثلها مثل أي لغات لها قواعد في الكتابة وتركيب خاص للجملة لكي تكون مفهومة (معروفة) من قبل مترجم لغة كوبول الذي وضع بناء على هذه القواعد والأسس. وتتكون اللغة من ستة عناصر أساسية وهي :

١ ـ	الكلمات المحجوزة	Reserved Words
٢ ـ	أسماء المتغيرات	Programmer - Supplied names
٣ ـ	الثوابت	literals
٤ ـ	الرموز (الإشارات) الخاصة	Symbols
٥ ـ	أرقام المستوى	Level numbers
٦ ـ	وصف الحقل (أو المتغير)	Picture

وكل تعليمة* ترد في هذه اللغة يجب أن تحتوي على الأقل كلمة محجوزة لتعطي المعنى المراد من هذه التعليمة أو العبارة. وهذه الكلمات يجب أن تذكر دون أدنى خطأ في التهجئة لأنها الأساس الذي يعتمد عليه مترجم اللغة أثناء قراءته للبرنامج. كما أن هذه الكلمات تختلف من مترجم لآخر ويجد القارىء في الملحق (١٥ ـ ٢) من هذا الكتاب قائمة بهذه الكلمات. كما يجب على المبرمج المبتدىء الانتباه الى هذه الكلمات والرجوع اليها لسببين :

أ ـ للتأكد من صحة كتابتها.

ب ـ لتجنب استخدامها كأسماء للحقول أو المتغيرات المراد تسميتها في البرنامج.

أما بالنسبة لأسماء المتغيرات وهي العنصر الثاني فهي عبارة عن مسميات يطلقها المبرمج على الأماكن (الحقول) المخصصة في الذاكرة الرئيسية للحاسب و يستخدمها

(*) يقصد بالتعليمة كلمة Statement أو Instruction

المبرمج في برنامجه ومعالجته للبيانات الخاصة بها وهذه المسميات تشمل :

- أسماء الفقرات في قسم الاجراءات.
- أسماء أماكن تخزين البيانات (الحقول، السجلات).
- أسماء الملفات المستخدمة في البرنامج.

وكل واحد من هذه المسميات يعتبر بمثابة عنوان* لمكان مخصص في الذاكرة يمكن المبرمج من الاشارة اليه لاستخدام محتواه في المعالجة. مثل PRINT - F تمثل اسم ملف الطباعة في البرنامج المذكور في الفقرة السابقة وكذلك PRINT - LINE اسم سطر الطباعة (سجل الطباعة) و START - P اسم للفقرة في قسم الاجراءات. (الشكل رقم ١ - ٣).

وعندما يختار المبرمج أسماء متغيرات برنامجه يجب عليه أن يراعي القواعد التالية :

١ - يمكن أن يحتوي الاسم على الأحرف من A الى Z والأرقام من (صفر) ∅ إلى 9 (تسعة) وإشارة الشرطة (-)** فقط و بدون أي فراغ بين الأحرف والأرقام المكونة للاسم.

٢ - يجب أن يحتوي الاسم على حرف واحد على الأقل ما عدا أسماء الفقرات والفصول*** يمكن أن تكون أرقاماً فقط.

٣ - لا يمكن أن يتكون أول رمز أو آخر رمز من الاسم إشارة الشرطة.

٤ - لا يمكن استخدام أسماء الكلمات المحجوزة كأسماء متغيرات.

٥ - يجب أن لا يزيد حجم الاسم عن ٣٠ رمزاً ****.

(*) tag

(**) Hyphen

(***)Pragraph and Section names

(****) قد يوجد بعض المترجمات تقبل حجماً أكبر من هذا الحجم القياسي.

والأمثلة التالية توضح كيفية استخدام هذه القواعد في وضع أسماء المتغيرات :

اسم المتغير	التوضيح
NAME	اسم مقبول
DIVISION	اسم غير مقبول لأنه يمثل كلمة محجوزة
DIVISION - 1	اسم مقبول
DIVISION - OF -	اسم غير مقبول لوجود شرطة في نهايته
DIVISION - OF - X	اسم مقبول
12345	اسم مقبول لفقرة أو لفصل ولكن غير مقبول كاسم لمتغير آخر

ويجب التنويه أن جميع الأسماء تكتب بالأحرف الكبيرة ولا تقبل الأحرف الصغيرة في أسماء المتغيرات.

أما العنصر الثالث في اللغة وهو الثوابت. و يقصد بالثابت قيمة محددة حرفياً أو ثابتاً رقمياً* . وعلى ذلك فانه يمكن أن يكون رقمي (عددي) مثل 1983 أو 261.56 أو -1ϕ.

أما اذا كان الثابت يمثل قيمة غير رقمية فانه يظهر في لغة كوبول بين إشارتين من الاشارة ' (الفاصلة العلوية)**.

وتعتبر هذه القيمة ثابتة في البرنامج ويمكن معالجتها أو مقارنتها مع حقول أخرى.

مثل :

MOVE 'INSTITUTE OF PUBLIC ADMINISTRATION' TO PRINT - LINE.

(*) Crawer and Crawford P. 22

(**) بعض المترجمات تستخدم الإشارة " بدلا ' (الفاصلة العلوية).

ووفق قواعد لغة كوبول لا يمكن أن يزيد حجم الثابت الرقمي على ١٨ رقماً ويمكن أن تبدأ من جهة اليسار بإشارة جبرية (سالبة أو موجبة) كما يمكن أن يحتوي الثابت على فاصلة عشرية ولكن يجب أن لا يكون آخر رمز فيه الفاصلة العشرية.

والعنصر الرابع في مكونات لغة كوبول هو العلامات الخاصة ويمكن تصنيفها الى ثلاثة أنواع :

- علامات تنقيط Punctuation
- علامات حسابية Arithmetic
- علامات العلاقة أو المقارنة Relational

و يوضح الشكل رقم (١ ـ ٤) هذه العلامات والغرض منها.

وتستخدم الفاصلة والفاصلة المنقوطة من أجل سهولة قراءة برنامج كوبول وحذفهما سوف لا يؤدي الى خطأ في الكتابة.

النوع	العلامة	الغرض منها
تنقيط Punctuation	.	تشير الى نهاية أي عبارة أو جملة في لغة كوبول
	,	للفصل بين العبارات في جملة كوبول
	;	
	' أو ''	لتوصيف الثوابت غير الرقمية
	( )	لاحتواء المؤشرات أو الأدلة أو التعبيرات الحسابية
حسابية Arithmetic	+	للجمع
	—	للطرح
	*	للضرب
	/	للتقسيم.

النوع	العلامة	الغرض منها
علاقة أو مقارنة Relational	**	للرفع الى قوة
	=	مقارنة المساواة
	>	لمقارنة أكبر من
	<	لمقارنة أصغر من

الشكل رقم (١ - ٤)
العلامات الخاصة في لغة كوبول

أما النقطة Period فذات أهمية كبيرة في كتابة جمل وعبارات لغة كوبول وحذفها قد يؤدي الى حدوث أخطاء كبيرة منطقية وشكلية. وعلامات التنقيط تخضع الى قاعدتين أثناء ترجمة برنامج كوبول :

- ضرورة مطلقة لوجود علامة تنقيط (مثل النقطة أحياناً) وعدم وجودها يؤدي الى حدوث خطأ أثناء الترجمة.
- ضرورة غير مطلقة لوجود علامة تنقيط وعدم وجود هذه العلامة لا يؤدي الى حدوث خطأ في الترجمة بل يقود الى خطأ في تنفيذ البرنامج (خطأ منطقي).

مثل وجوب وجود فراغ تالي لأ ية فاصلة أو فاصلة منقوطة أو نقطة وكذلك يجب إنهاء جميع عبارات أو جمل لغة كوبول Entries بنقطة.

أمثلة :

```
1.   INPUT-OUTPUT SECTION
2.   OPEN INPUT CARD-FILE,M-FILE.
3.   OPEN OUTPUT PRINT-F.
```

ففي 1 و 2 جملة غير مقبولة الأ ولى لعدم وجود نقطة في نهاية العبارة والثانية لعدم وجود فراغ بعد الفاصلة.

وسوف يتم شرح العلامات الحسابية والعلائقية بالتفصيل في الجزء التالي من الكتاب .

أما العنصر الخامس من لغة كوبول فهو أرقام المستوى و يقصد بها الأ رقام التي توضح علاقة الحقول الجزئية بالحقول الاجمالية وكيفية تجزئتها . فمثلا وصف سجل بطاقة ما يأخذ الشكل التالي :

```
ϕ1  CARD-REC.
    ϕ2  CARD-NAME      PICTURE IS A(25).
    ϕ2  CARD-NUMBER    PICTURE IS 9(5).
    ϕ2  CARD-ADDRS.
        ϕ5  CARD-CITY      PICTURE IS A(12).
        ϕ5  CARD-STREET    PICTURE IS X(12).
    ϕ2  FILLER     PICTURE IS X(16).
```

والرقم ϕ1 يعني رقم المستوى الأ ول و يعني أن CARD-REC اسم الحقل (مكان التخزين) اجمالي يحتوي على حقول جزئية تأخذ رقم المستوى ϕ2 , ϕ5 , ... الى غاية الرقم 49 وكذلك الحقول الموصوفة بالرقم ϕ5 هي حقول جزئية للحقل CARD-ADDRS'.

وسوف يأتي شرح أكثر تفصيلاً لذلك في الفصول القادمة .

أما العنصر السادس وهو وصف الحقل بعبارة PICTURE* التي تستخدم لتحديد شكل الحقل وطبيعة البيانات المحتواة فيه سواء كان مخصصاً لغرض ادخال هذه البيانات أو اخراجها وسوف نأتي لذكر هذه العبارة بصورة أكثر تفصيلا فيما بعد، ولكن يمكن ذكر ثلاثة أنواع أساسية لوصف الحقل وتحديد نوع محتواه وهي الرمز 9

(*) وتترجم بالعربية الى صورة (أي شكل الحقل وطريقة توصيفه) .

و يعني أن الحقل الموصوف يأخذ شكل رقمياً أما الرمز A فيعني أن الحقل أبجدي الشكل. أما الرمز X فيدل على أن الحقل يمكن أن يحتوي على كلا النوعين، أبجدي ورقمي مع امكانية احتوائه للإشارات الخاصة.

١ ـ ٥ صفحة ترميز برنامج كوبول COBOL CODING FORM

يتم كتابة تعليمات وعبارات لغة كوبول على ورق معين (خاص) صمم وفقاً لقواعد كتابة كوبول وفق الشكل رقم (١ ـ ٥) و يبلغ طول السطر في الصفحة (٨٠) عموداً أو موضعاً لكتابة تعليمات وجمل لغة كوبول و يُحوّل كل سطر في هذه الصفحة الى بطاقة مثقبة في حال ادخال البرنامج بوساطة البطاقات المثقبة أما عند استخدام منقح النصوص TEXT EDITOR فإن كل سطر في الصفحة يعادل سطرا واحدا على الشاشة المرئية.

وفي أعلى الصفحة يوجد مكان مخصص لكتابة الاشياء التعريفية العامة بالبرنامج والنظام واسم المبرمج وتاريخ كتابة البرنامج واسم النظام واسم البرنامج وتعليمات إدخال (أو تثقيب) البرنامج الخاصة وشكل البطاقة المراد تثقيبها (يوجد نوعين من أشكال التثقيب A و B معروفين عالمياً). وكذلك رقم الصفحة في أقصى اليمين من الورقة.

وتعليمات إدخال أو تثقيب البطاقات الخاصة بالبرنامج خاصة عندما يطلب المبرمج من عامل إدخال البيانات القيام بهذه المهمة. ولكي يتجنب المبرمج أي خطأ في عملية الإدخال عليه أن يوضح للشخص الآخر الحروف والأرقام التي يوجد بينها تشابه وأن يميز بينها: مثل الحرف o والرقم ∅ صفر الذي يوضع عليه الاشارة / وكذلك الحرف I والرقم 1 والحرف S والرقم 5. فيمكن للمبرمج توضيح ذلك في الجزء العلوي من الصفحة المخصص لهذا الغرض.

ولتسهيل مهمة عامل ادخال البيانات يكتب عادة البرنامج بشكل يتخلله أسطر فارغة (سطرين كتابة بينها سطر فارغ).

والجزء المهم في صفحة الترميز هو الذي يلي الجزء العلوي حيث أن كل سطر مقسم الى (٨٠) عمود وكل عمود يقابل عمود في البطاقة المثقبة. وتوجد أرقام لهذه الأعمدة في أعلى الصفحة وفي أسفلها لتسهيل احتساب عدد الأعمدة في السطر على المبرمج. وكل عمود (او موضع) مخصص لحرف أو رقم أو فراغ أو اشارة خاصة.

ونـنتقل الآن الى السطر الواحد حيث أن الستة المواضع (الأعمدة) الأولى مخصصة للرقم المتسلسل للسطر وقد كانت أهمية هذا الرقم عالية جداً عند استخدام البطاقات المثقبة لادخال البرنامج بينما انخفضت أهميته بعد استخدام منقح النصوص لتسجيل البرنامج. حيث أنه يحفظ تسلسل الأسطر (البطاقات) وفق التسلسل المنطقي الموضوع للبرنامج، وعادة هذا الرقم يقسم الى جزئين كل جزء مؤلف من ثلاثة أعمدة الأول من العمود ١ الى ٣ مخصص لرقم الصفحة والثاني من العمود ٤ الى ٦ مخصص لرقم السطر ضمن الصفحة.

أما العمود رقم ٧ والذي يحمل العنوان Cont فانه يستخدم للاشارة الى أن بعض تعليمات كوبول تزيد عن السطر المخصص وتحتاج الى تتمة (Continuation) وتستعمل الاشارة (ـ) للدلالة على ذلك. أما اذا وضعت الاشارة (*) في هذا العمود فان ذلك يعني أن هذا السطر مخصص لاثبات ملاحظات وتعليق عن البرنامج وليس سطرا من ضمن أسطر التعليمات ولا يقوم الحاسب بتنفيذه أثناء تنفيذ البرنامج.

والجزء الفعلي المخصص لكتابة تعليمات وعبارات كوبول في السطر يكون من العمود رقم ٨ الى العمود رقم ٧٢ وهذه المساحة مقسومة الى قسمين A و B الاعمدة ٨ الى ١١ والاعمدة ١٢ الى ٧٢ على التتالي.

والسبب في هذا التقسيم الى جزئين A و B واضح و بسيط وذلك من أجل سهولة القراءة وتدوين البرنامج حيث أن النقاط الرئيسية والتي تعتبر علامة بارزة في البرنامج تبدأ من الجزء A وتدعى (KEY LINES) أما الأسطر التي تعتبر أقل أهمية فتبدأ في الجزء B. فمثلاً أسماء أقسام البرنامج والفصول التابعة لها تبدأ من الجزء A وكذلك أرقام ومستويات توصيف البيانات. وفي قسم الاجراءات تكتب كافة أسماء الفقرات (PARAGRAPHS) ابتداءً من الجزء A. أما بقية الأسطر والتعليمات فتبدأ من الجزء B.

وأخيراً يوجد جزء معنون باسم (IDENTIFICATION) و يأخذ الأعمدة ٧٣ الى ٨٠ من السطر فهو مخصص لاضافة معلومات اختيارية عن هوية السطر وعادة يدون فيها أحرف خاصة باسم المبرمج أو البرنامج مع أرقام متسلسلة أو أرقام مميزة للبرنامج أو الفقرات ضمن البرنامج. وهذا الجزء حسب قواعد لغة كوبول لا يُأخذ بعين الاعتبار أثناء ترجمة (COMPILATION) البرنامج الى لغة الهدف (OBJECT)* او صيغة التنفيذ.

إن وجود صفحة خاصة لترميز برنامج كوبول لا يعني عدم إمكانية كتابة تعليمات وعبارات كوبول على ورق أو صفحات عادية ولكن شكل هذه الصفحة يسهل الكتابة وفق القواعد الموضوعة للغة. وظهور الشاشة المرئية واستخدام منقح النصوص قلل من أهمية استخدام هذا الشكل المخصص للكتابة، ولكنه لا زال استخدامه أمراً ضرورياً خاصة في مراحل تعليم لغة كوبول الأولية.

(*) لغة الآلة.

صفحة ترميز لغة كوبول

COBOL Coding

CARD FORM

1 — 7	A 8 —	B 12 — COBOL STATEMENT	73 —

الشكل رقم (١ - ٥)

١ - ٦ تمارين

١ - بين فيما اذا كانت هذه العبارات صحيحة أم خطأ ؟

- الثوابت غير الرقمية يمكن أن لا تحتوي أرقاماً .
- استعمال الأعمدة ٧٣ الى ٨٠ في صفحة الترميز اختياري .
- تظهر أسماء الملفات في ثلاثة أقسام DIVISIONS .
- العمود رقم ٨ يستخدم للدلالة على استمرارية التعليمة أو الجملة .
- الثابت الرقمي يمكن أن يحتوي على ١٨ رمزاً .
- يمكن أن تظهر الكلمات المحجوزة Reserved Words في الثوابت غير الرقمية .

٢ - بين أي من هذه الأسماء الموضوعة لتسمية الحقول في البرنامج مقبولة وصحيحة وإذا كان الاسم غير مقبول وضح سبب ذلك :

a) NUMBER - OF - STUDENTS

b) 1 2 3 4 5 6

c) FIVE SIX

d) IDENTIFICATION - DIVISION

e) STORE -

٣ - أشر طبقاً لقواعد كتابة لغة كوبول أي من هذه التعليمات يعتبر مقبولاً وصحيحاً وأي منها خطأ :

a) OPEN INPUT R-FILE, Z-FILE.

b) MOVE B TO C.

c) MOVESPACE TO T1, T2.

d) MOVE 'A' TO X.

e) WRITE RECORD - IN AFTER TWO LINES.

f) CLOSE IN-FILE/PRINT-FILE.

g) CLOSE CARD-FILE PRINT-FILE.

الفصل الثاني

التعليمات الأولية في لغة كوبول

(قسم الاجراءات)

٢ ـ ١ لمحة عامة

إن قسم الاجراءات PROCEDURE DIVISION في برنامج لغة كوبول هو الجزء الذي يحتوي على المنطق في البرنامج، وهو القسم الذي يقوم عملياً بأداء الغرض من البرنامج، وسوف نحاول تغطية التعليمات الأولية والمهمة في لغة كوبول ضمن هذا الفصل. وتشمل هذه التعليمات عمليات إدخال وإخراج البيانات من والى البرنامج (أي الحاسب الآلي) وهي : OPEN, READ, WRITE, CLOSE, وكذلك التعليمات التي تقوم بأداء العمليات الحسابية ADD, SUBTRACT, MULTIPLY, DIVIDE, COMPUTE وتعليمة MOVE التي تنقل البيانات من المناطق المختلفة ضمن الذاكرة الرئيسية في الحاسب وتعليمة المقارنة IF التي تمكن المبرمج من الاختيار بين أكثر من اتجاه في سير البرنامج وتعليمة PERFORM التي تنفذ عملية الدوارة* ثم تعليمة إنهاء البرنامج بصورة منطقية وهي STOP RUN.

٢ ـ ٢ قراءة صيغ لغة كوبول

لقد تم وضع أشكال عامة لمجموعة صيغ لغة كوبول ويمكن قراءة هذه الأشكال وفق الأسس التالية :

(*) LOOP

١ ـ تظهر الكلمات المحجوزة ضمن الشكل العام بالأحرف الانكليزية الكبيرة

٢ ـ يوضع خط تحت الكلمات المحجوزة التي يجب أن توجد في العبارة أو الجملة ، أما الكلمات الاختيارية Optional فلا يوضع خط تحتها .

٣ ـ تشير الكلمات المكتوبة بالأحرف الصغيرة الى أسماء المتغيرات التي تعطى من قبل المبرمج .

٤ ـ تشير الأقواس المربعة [ ] الى العبارات أو الكلمات الاختيارية في التعليمة أو الجملة .

٥ ـ تشير الأقواس { } إلى أن أحد البنود المذكورة ضمنها الذي يجب اختياره أو ذكره في العبارة .

٦ ـ تشير النقاط (...) الى امكانية تكرار المقطع الأخير من العبارة عددا من المرات* .

ويمكن توضيح هذه العلامات أو الرموز في المثال التالي الذي يوصف جزءا من تعليمة IF :

```
    { identifier - 1 }   { IS [NOT] GREATER THAN }   { identifier - 2 }
IF  {    literal - 1  }   { IS [NOT] LESS THAN    }   { liferal - 2    }
    {                 }   { IS [NOT] EQUAL TO     }   {                }
```

فكلمة IF اجبارية في التعليمة لأنه يوجد خط تحتها وكذلك الكلمات NOT, EQUAL, GREATER, LESS ، أما كلمة Identifier - 1 وكلمة Literal - 1 فيذكر أحداهما في التعليمة ، وهما تمثلان الاسم أو القيمة التي يحددها المبرمج . بينما الكلمات IS, THAN, TO فهي اختيارية في التعليمة هذه .

(*) عدد مرات تكرار أي مقطع تحكمي Arbitrary و يضبطه نوع المترجم المستخدم و يذكر في مراجع الشركة الصانعة للحاسب عادة .

٢ ـ ٣ التعليمات الأولية للمعالجة ونقل التسلسل :

وتشمل هذه التعليمات ما يلي :

- تعليمات القراءة والكتابة وفتح وغلق الملف :

OPEN, READ, WRITE, CLOSE

- تعليمة نقل وتحريك البيانات :

MOVE

- تعليمات نقل تسلسل تنفيذ البرنامج والمقارنة والتوقف

GO TO, PERFORM, IF, STOP

٢ ـ ٣ ـ ١ تعليمة فتح الملف OPEN

كل ملف في برنامج كوبول يراد معالجته يجب ذكر تعليمة OPEN قبل هذه المعالجة. وتطلب هذه التعليمة من نظام التشغيل توفير الملف المعرّف على الوحدات المتصلة بالحاسب سواء كان هذا الملف مخصصا للقراءة أو للكتابة (التسجيل) عليه. وتأخذ هذه التعليمة الشكل العام التالي :

```
        { INPUT  }
OPEN    { OUTPUT }    file - name - 1 [,file - name - 2 ....]
        { I _ O  }
        { EXTEND }
```

وضمن هذه التعليمة يحدد المبرمج الغرض من استخدام الملف في البرنامج فكلمة INPUT تعني أن الملف تم فتحه لغرض القراءة أو لادخال البيانات منه للبرنامج وكلمة OUTPUT تعني أن الملف سوف يستخدم لغرض التسجيل عليه، أما كلمة

I - O فهي تستخدم لوصف الملف لأجل عمليات القراءة والكتابة معاً في نفس البرنامج أما كلمة EXTEND وتعني توسيع حجم الملف وتستعمل لغرض الاضافة على ملف سابق سجل على شريط ممغنط بشكل خاص. ويمكن لتعليمة OPEN أن تحتوي أكثر من نوع بنفس الوقت لاكثر من ملف.

أمثلة :

1. OPEN INPUT CARD-FILE.
2. OPEN INPUT CARD-FILE, OUTPUT PRINT-F.
3. OPEN I - O SCRATCH - F.
4. OPEN EXTEND MTP-FILE.

٢ ـ ٣ ـ ٢ تعليمة قراءة البيانات READ

تستخدم تعليمة READ لقراءة البيانات من خارج برنامج كوبول (أي من وحدات الإدخال والإخراج المتصلة مباشرة بالحاسب) لكي تتم معالجتها في البرنامج. وتأخذ هذه التعليمة الشكل العام المختصر التالي :

<u>READ</u> file - name [AT <u>END</u> imperative - statement]

حيث أن اسم الملف المراد قراءته يرد بعد كلمة READ وتتم القراءة من الملف على أساس سجل واحد لكل تعليمة قراءة. أما عبارة AT END الاختيارية فهي تمثل شرطا يتحقق عندما تصل عملية القراءة الى نهاية الملف ولم يعد يوجد سجلات لأجل القراءة عندئذ ينتقل تسلسل التنفيذ الى التعليمة الواردة مباشرة بعد كلمة END.

مثال :

READ CARD - FILE AT END MOVE ' NO ' TO MORE - DATA

ففي هذا المثال تقوم تعليمة القراءة بقراءة سجل من الملف المسمى CARD - FILE وعندما تصل القراءة (في حال تكرارها من قبل المبرمج في برنامجه) الى نهاية الملف تتم

عملية نقل القيمة الثابتة NO الى الحقل المسمى MORE - DATA . أما إذا لم يتحقق شرط لنهاية الملف فيتابع الحاسب تنفيذ التعليمة التالية لتعليمة READ مباشرة.

ويجب التأكيد على أن تعليمة READ تأخذ اسم الملف المقروء الى جانبها وليس اسم السجل.

٢ - ٣ - ٣ تعليمة نقل البيانات MOVE

تقوم تعليمة النقل MOVE في لغة كوبول بعملية نقل البيانات (قيمة الحقل) وتحريكها بين أماكن التخزين في الذاكرة، وهذا النقل لا يعني النقل الفيزيائي للشيء المنقول بل هو عبارة عن نسخ للمحتوى و يبقى نفس المحتوى في حقل مصدر البيانات. وشكل تعليمة النقل العام كما يلي :

```
         { Identifier - 1 }
MOVE     {                }  TO Idenfifier - 2 [,    Identifier - 3] ....
         { Literal        }
```

حيث تعني كلمة Identifier - 1 اسم الحقل أو مكان التخزين في الذاكرة والموصوف من قبل المبرمج في قسم البيانات. وكلمة Literal تعني القيمة الحرفية أو الثابتة المراد نقلها (أو بالأحرى نسخها).

أمثلة :

1) MOVE 205 TO P - O - BOX.

2) MOVE 'RIYADH' TO CITY - NAME.

3) MOVE STUDENT - NAME TO PRINT - NAME.

ففي المثال الأول تقوم تعليمة النقل بوضع القيمة 205 في الحقل P - O - BOX أما المثال الثاني فتتم عملية نقل كلمة RIYADH الى الحقل CITY - NAME أما الثالث

فيتضمن نقل محتوى (نسخ محتوى) الحقل STUDENT - NAME الى الحقل مستقبل البيانات وهو PRINT - NAME.

ويمكن التمييز في عملية نقل البيانات MOVE ضمن لغة كوبول بين نوعين من البيانات :

١ ـ البيانات غير الرقمية : وتتم عملية النقل (النسخ) بين حقل مصدر البيانات SOURCE والحقل مستقبل البيانات RECEIVING على أساس رمز واحد من البيانات في كل مرة وعلى التوالي من جهة اليسار الى اليمين. وإذا كان حقل مستقبل البيانات أكبر من حقل المصدر يُملأ الجزء الفائض من هذا الحقل بعد عملية تنفيذ النقل بفراغات BLANKS أما إذا كان الحقل المستقبل أصغر حجماً من حقل المصدر فان الجزء الزائد من جهة اليمين يُحذف TRUNCATED ولا تتم عملية نسخه.

مثال : MOVE A TO B.

قبل التنفيذ :	B : ؟ ؟ ؟ ؟ ؟ ؟ ؟	A : A B C D E
بعد التنفيذ :	B : A B C D E _ _	A : A B C D E

MOVE C TO D.

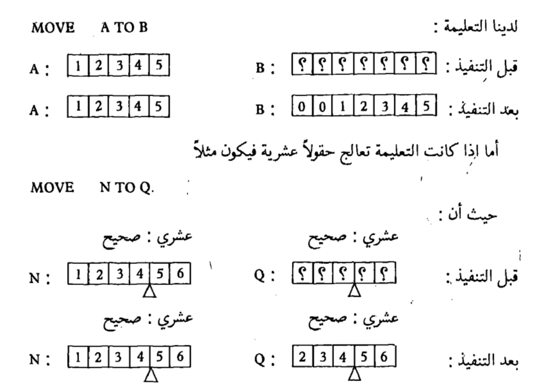
قبل التنفيذ :	D : ؟ ؟ ؟ ؟	C : A B C D E F
بعد التنفيذ :	D : A B C D	C : A B C D E F

٢ ـ البيانات الرقمية : تتم عملية النسخ هنا من جهة اليمين الى اليسار وتتم رقم رقم digit في حال عدم وجود فاصل عشرية في الحقل المنقول (المنسوخ).

أما اذا كان الحقل يحتوي على قيمة رقمية عشرية فتكون الفاصلة العشرية هي بمثابة النقطة المرشدة في كلا الحقلين المصدر والمستقبل للبيانات.

وإذا كان حقل مستقبل البيانات أكبر من حقل مصدر البيانات فتتم عملية إضافة أصفار الى يسار الرقم المنقول (المنسوخ) لملء الجزء الفائض بعد النقل. أما اذا كان العكس فيحدث حذف للجزء الزائد عن حجم الحقل المستقبل من جهة اليسار (أي القيم العليا والكبيرة في الحقل الرقمي أو القيمة الثابتة المراد نقلها).

مثال :



وتجدر الاشارة هنا الى أن الفاصلة العشرية Decimal Point لا تمثل مكاناً خاصاً (Byte) بها وإنما يشار اليها بوساطة إشارة خاصة في الذاكرة* و يرمز لها عند تمثيل الحقل بالرمز (△).

(*) سوف يرد في الفصل القادم كيفية توصيف الفاصلة العشرية وتفاصيل أخرى عنها.

كما يجب التنويه أيضاً الى أن حجم الحقول الرقمية في عملية المناقلة (MOVE) يجب أن لا يزيد عن ١٨ رمزاً لكل حقل مصدر بيانات بما فيها الإشارة الجبرية. أما الحقول غير الرقمية فيمكن أن تصل الى حجم كبير ويختلف باختلاف الحاسب المستخدم لتنفيذ برنامج كوبول.

٢ ـ ٣ ـ ٤ تعليمة كتابة السجل WRITE

تستخدم تعليمة الكتابة WRITE لكتابة أو تسجيل سجل ما على ملف الاخراج بوساطة وحدات الاخراج، وتأخذ هذه التعليمة الشكل العام التالي :

```
WRITE record - name [ { AFTER  } ADVANCING { identifier } [ { LINE  } ] ]
                        { BEFORE }            { integer    }   { LINES }
```

حيث أن عبارة ADVANCING اختيارية وتقوم بوظيفة ضبط عدد الاسطر عند تسجيل السجلات (الاسطر) على الطابعة لملف اخراج النتائج من البرنامج. وإذا حذفت هذه العبارة فتتقدم الطابعة سطرا واحدا فقط. وكلمة AFTER تعني الكتابة بعد تقدم عدد من الأسطر.

مثال :

WRITE P - LINE AFTER 3 LINES

ففي هذه التعليمة تتم كتابة السجل P - LINE بعد تقدم الطابعة سطرين فارغين وتتم الكتابة على السطر الثالث. أما كلمة BEFORE فتستخدم لكتابة السجل أو السطر قبل تقدم الطابعة عدد الأسطر المذكور في هذه التعليمة.

ويجب التأكيد هنا الى أن تعليمة WRITE تستخدم اسم السجل المكرر في الملف سواء كان هذا الملف يكتب على وسيط مغناطيسي أم على طابعة. بينما تستخدم تعليمة القراءة READ اسم الملف من أجل قراءة سجل واحد من هذا الملف.

٢ ـ ٣ ـ ٥ تعليمة غلق الملف CLOSE

يجب غلق الملفات المستخدمة في برنامج كوبول بعد الانتهاء من معالجتها وتقوم تعليمة CLOSE بهذه الوظيفة وهذه التعليمة ضرورية خاصة عندما يكون الملف المستخدم ملف اخراج لأنه بعد تنفيذ عملية غلق الملف يضمن المبرمج سلامة السجلات في الملف. وتأخذ هذه التعليمة الشكل العام التالي :

```
CLOSE     file - name - 1 [,file - name - 2]....
```

حيث أن file - name -1 تمثل اسم الملف المستخدم في البرنامج (عادة يكون الاسم المنطقي للملف). و يلاحظ أن هذه التعليمة تقوم بعملية غلق ملف أو أكثر ولا تحتوي على نوع الملف (مثل INPUT, OUTPUT, I - O).

مثال :

```
CLOSE CARD - FILE.
CLOSE CARD - FILE, PRINT - FILE.
```

وتجب الاشارة الى أن تعليمة غلق الملف ليس بالضرورة أن ترد في آخر البرنامج بل يمكن أن تكون في أي مكان وحسب وصفها المنطقي والحاجة اليها، كما يمكن إعادة فتح نفس الملف بعد غلقه بهذه التعليمة بوساطة تعليمة OPEN تالية لها.

٢ - ٣ - ٦ تعليمة نقل التسلسل GO TO

تمكن تعليمة «اذهب الى» GO TO المبرمج من نقل تسلسل تنفيذ البرنامج في قسم الاجراءات الى فقرة أخرى. وتعطيه مرونة في التنقل واختيار التعليمات المراد تنفيذها وفق تسلسل البرنامج المنطقي. وتأخذ هذه التعليمة الشكل العام التالي :

GO TO [Procedure - name]

حيث يقصد باسم الاجراء Procedure - name اسم الفقرة المراد القفز اليها بعد تنفيذ GO TO.

مثال :

GO TO READ - RECORD.

GO TO NEXT - PARAGRAPH.

حيث أن READ - RECORD ;NEXT - PARAGRAPH أسماء فقرات (أو اجراءات) معرفة في قسم الاجراءات ضمن البرنامج.

ويجب التنويه هنا الى أن اسلوب البرمجة الحديث* يتجه نحو التقليل ما أمكن من هذه التعليمة ضمن البرنامج لأنها من السهل أن تكون سبباً في حدوث اخطاء منطقية بسبب عملية النقل التي تؤديها بين الفقرات ووجودها الكثير ضمن البرنامج يجعله متشابكاً جداً مما يصعب تتبعه وتدقيقه.

٢ - ٣ - ٧ تعليمة التوقف STOP

كل برنامج كوبول يجب أن يحتوي على تعليمة توقف واحدة على الأقل** وتدل هذه التعليمة على النهاية المنطقية لتنفيذه. وتأخذ هذه التعليمة الشكل العام :

(*) Structured Programming

(**) إذا لم يمكن هذا البرنامج معرف كبرنامج فرعي Subprogram

```
STOP  { RUN     }
      { literal }
```

حيث أن كلمة RUN عندما ترد مع كلمة STOP تؤدي الى توقف البرنامج عن التنفيذ وتدل على نهايته المنطقية و ينتقل نظام التشغيل بعد ذلك لتنفيذ عمل آخر.

أما اختيار اضافة ثابت حرفي Literal الى جانب كلمة STOP فيتيح إظهار رسالة خاصة لمشغل الحاسب ليقوم بالاجابة بعد اطلاعه عليها. و بذلك تعطي هذه التعليمة عندئذ امكانية تدخل مشغل الحاسب عند الطلب وبمعنى آخر تؤدي هذه التعليمة في تلك الحالة الى توقف مؤقت PAUSE ريثما يعطي المشغل اجابته على الرسالة المظهرة. كما تجب الاشارة الى أن هذه التعليمة ليس بالضرورة أن تكون آخر تعليمة في تسلسل البرنامج بل توضع عند النهاية المنطقية للبرنامج ضمن قسم الاجراءات.

مثال :

```
STOP RUN.
STOP 'IF NO MORE TAPES ENTER 1'.
```

٢ - ٣ - ٨ تعليمة نفذ (أنجز) PERFORM

تعتبر تعليمة PERFORM من التعليمات الأكثر أهمية وفعالية في لغة الكوبول. فهي تمكن المبرمج من نقل تسلسل تنفيذ البرنامج من فقرة (Paragraph) إلى فقرة أخرى أو أي فقرة في مكان ما ضمن البرنامج. وتكمن أهمية هذه التعليمة في امكانية تقسيم أي برنامج كوبول متشابك أو معقد الى أجزاء سلسة وسهلة و واضحة بحيث يمكن صياغة الجزء الرئيسي من البرنامج مشتملاً على عدة تعليمات PERFORM متسلسلة

وكل تعليمة منها تؤدي أو تنجز فقرة أو مجموعة فقرات من بقية البرنامج (أي الأجزاء الفرعية).

وتبرز أهمية استخدام هذه التعليمة عند اللجوء الى تطبيق البرمجة الهيكلية (Structured Programming) في كتابة برنامج كوبول، وسنبحث ذلك في حينه.
وتعليمة نفذ تأخذ أشكالا مختلفة عامة وأبسطها الشكل التالي :

PERFORM Procedure - name

وعبارة Procedure - name يقصد بها اسم "الأجراء" في قسم الاجراءات أو اسم فقرة من هذا القسم. وفعل التعليمة يطلب من الحاسب تنفيذ هذا الاجراء عند وصول تسلسل تنفيذ تعليمات البرنامج الى هذه التعليمة.

مثال :

```
     OPEN OUTPUT PRINT - FILE, INPUT IN - FILE.
     PERFORM WRITE - HEADER - LINES.---->(1)
       (2)---->
READ - NXT.
     READ IN - FILE AT END PERFORM END - ROUTINE,
     ------>(3)(4)-->  STOP RUN.
(1)  MOVE FLD - 1 TO OUT - FLD - 1.
 |    |
 |    |
 |    |
 |   GO TO READ - NXT.
 |
 +-> WRITE - HEADER - LINES.
```

```
          MOVE SPACE TO PRINT - LINE.
          WRITE PRINT - LINE AFTER PAGE.
          MOVE DAADER - LN1 TO PRINT - LINE.
          WRITE PRINT - LINE AFTER 3.
 (3)      MOVE SPACE TO PRINT - LINE
  |       WRITE PRINT - LINE AFTER 1.
  |               ---------> (2)
  └> END - ROUTINE.
            CLOSE IN - FILE, PRINT - FILE.
                 ---------> (4)
```

فتعليمة PERFORM WRITE - HEADER - LINES تقوم بنقل تسلسل البرنامج الى الفقرة (١) وبعد تنفيذ هذه الجزء يعود تسلسل البرنامج الى الجزء بعد تعليمة PERFORM وهي النقطة (٢)، أي ..READ ، وكذلك تعليمة PERFORM لجزء من الاجراءات المسمى END - ROUTINE وبعد تنفيذه يعود تسلسل البرنامج عند التنفيذ الى تعليمة STOP RUN .

وعندما تكون الفقرة أو جزء الاجراء المراد تنفيذه في تعليمة نفذ مؤلف من أكثر من فقرة فيكون شكل تعليمة PERFORM العام التالي :

PERFORM Procedure - name - 1 [THROUGH / THRU] Procedure - name - 2

وعبارة THRU الاختيارية تسمح بتنفيذ جزء الاجراءات المراد بتعليمة PERFORM من أول فقرة مسماة وهي Procedure - name - 1 الى الفقرة النهائية فيه وهي Procedure - name - 2 (ملاحظة : الفقرة الأخيرة تعليماتها مشمولة في الجزء المطلوب) .

وغالباً ما تكون أخر فقرة مكونة من اسم فقرة مضافاً اليها كلمة EXIT . وعبارة EXIT هذه هي تعليمة من تعليمات كوبول ولا تقوم بتنفيذ أي عمل ولكنها تعبر فقط عن نهاية جزء الاجراء المرغوب (المسمى) في تعليمة PERFORM .

وشكل تعليمة EXIT العام هو :

Paragraph - name. EXIT

ويجب ملاحظة أن اسم الفقرة السابقة لهذه العبارة ليس إجباريا فهو(*) اختياري ، لذلك فانها يمكن أن ترد في أخر أي فقرة نهائية للجزء الموصف في تعليمة PERFORM .

وتجدر الملاحظة أنه على المبرمج مراعاة عودة تسلسل تنفيذ البرنامج بعد الجزء المسمى في PERFORM الى النقطة التالية لهذه التعليمة . لهذا السبب يمكن أن تكون عبارة EXIT نقطة إنهاء الجزء المطلوب تنفيذه بتعليمة PERFORM . كما يجب التنويه هنا الى امكانية استخدام عبارة GO TO ضمن الجزء المستدعى في PERFORM على أن لا تسبب نقل التسلسل الى خارج الجزء المحدد لذلك .

مثال :

PERFORM READ - PROCESS THRU E - READ - PROCESS.

(*) الحاسب TI990 يتطلب اسم فقرة لكل تعليمة EXIT .

```
→ READ - PROCESS.
        READ IN - FILE AT END GO TO E - READ - PROCESS.
  MOVE - PROCESS.
        MOVE FLD - 1 TO W - FLD - 1.
        MOVE FLD - 2 TO W - FLD - 2.
→ E - READ - PROCESS.
        CLOSE IN - FILE.
  WRITE - TOTALS .
        |
        |
```

مثال :

```
        PERFORM READ - CARDS THRU E - READ -'CARDS.

→ READ - CARDS.
        READ CARD - F AT END CLOSE CARD - F GO TO
        |
        |                                  E - READ - CARDS'.
        |
→ E - READ - CARDS.
        EXIT.
```

و يسمى المجال ما بين نقطة (فقرة) البداية في الجزء المطلوب ونقطة (فقرة) النهاية بمجال PERFORM أو

- Range of PERFORM -

ويمكن ان تكون مجالات تنفيذ PERFORM متعددة في البرنامج الواحد وتجب عندئذ الإشارة الى تجنب أي تداخل في هذه المجالات لأنه بسبب خطأ منطقيا في سير تنفيذ البرنامج .

مثال صحيح :

عن مجالين متداخلين الأول كبير والثاني أصغر :

```
   A.  PERFORM  B  THRU F.
[  B.  ______ .
|  C.  PERFORM  D  THRU E.
| [D.  ______ .
| [E.  ______ .
[  F.  ______ .
```

مثال صحيح :

عن مجالين متعددين غير متداخلين :

```
   A.  PERFORM  B  THRU D.
[  B.  ______ .
|  C.  PERFORM  M  THRU Q.
[  D.  ______ .
   G.  ______ .
[  M.  ______ .
|  I.  ______ :
[  Q.  ______ .
```

مثال غير صحيح :

بسبب تداخل مجالي تعليمة PERFORM

```
     A.  PERFROM  B  THRU E.
  [  B.  ______ .
  |  C.
  |      PERFORM  D  THRU F.
[ |  D.  ______ .
| [  E.  ______ .
|    F.
[        ______ .
```

ولتنفيذ إجراء معين (فقرة أو فقرات معينة) عدد ثابت من المرات تأخذ تعليمة PERFORM الشكل التالي :

```
PERFORM Procedure - name - 1  [ THRU Procedure - name - 2 ]

   { identifier }
   { literal    }   TIMES
```

حيث يمثل المتغير أو الثابت رقماً صحيحاً* يدل على عدد المرات المراد تنفيذها للاجراء المذكور (وتمثل الدوارة Loop) . ثم يعود تسلسل تنفيذ البرنامج الى طبيعته بعد الانتهاء من عدد المرات المذكور في تعليمة PERFORM والممثل في قيمة المتغير أو الثابت literal . identifier,

مثال :

```
        MOVE 5 TO N.
        PERFORM PARA - 1 THRU PARA - 10 N TIMES.
        |
        |
        |
PARA - 1.     MOVE ZERO TO TOT - FLD.
        |
        |
        |
PARA - 10.    EXIT.
```

(*) أكبر من الصفر وغير سالب، وفي حال كون هذه القيمة صفراً أو عدداً ثابتاً يتم تنفيذ التعليمة التالية لـ PERFORM.

تعليمة نفذ المشروطة :

تأخذ تعليمة PERFORM شكلاً آخر عند استخدام العبارة UNTIL معها :

PERFORM Procedure - name - 1 [THRU Procedure - name - 2]
[UNTIL Condition]

وهذا الشكل العام من التعليمة يضيف شرطاً Condition اليها فإذا تحقق هذا الشرط قبل الوصول الى تنفيذها (تتم المقارنة عليه قبل تعليمة PERFORM) فان الاجراءات المطلوبة (الفقرات الموصوفة) في التعليمة نفذ سوف لا تتم ولا ينتقل تسلسل البرنامج اليها، و يتابع الحاسب تنفيذ البرنامج وفق التسلسل العادي له (أي التعليمة التي تلي PERFORM مباشرة). وإذا لم يتحقق الشرط الموصوف فإن الحاسب ينفذ الاجراءات المطلوبة عدداً من المرات و يتأكد في كل مرة من أن الشرط الموصوف قد تحقق أم لا. وعند تحقق الشرط ينتقل التسلسل الى التعليمة التالية مباشرة.

مثال :

```
MOVE 10 TO N.
PERFORM P - AA THRU P - EE UNTIL N IS
EQUAL 10. *
```

وفي هذا المثال فان الحاسب سوف لا ينفذ تعليمة PERFORM لأن الشرط تحقق قبل الوصول اليها (أي أن N = 10).

(*) عبارة N = 10 أو عبارة IS EQUAL مستخدمة في لغة الكوبول.

مثال :

```
        MOVE SPACE TO PRINT - LINE.
        MOVE 1 TO N.
        PERFORM PARA - A UNTIL N = 5. *
        |
        |
        |
PARA - A.
        WRITE PRINT - LINE AFTER 1.
```

إن الـفـقرة PARA - A سوف تنفذ ٤ مرات وليس ٥ مرات و بعد المرة الرابعة تكون قيـمـة N مـساو ية للرقم 5 . حيث أن مقارنة القيمة N تتم قبل البدء في تنفيذ التعليمة PERFORM .

مثال :

```
        MOVE ' NO ' TO END - OF - FILE - FLD.
    PERFORM READ - CARDS UNTIL END - OF - FILE - FLD IS EQUAL ' YES '.
        |
        |
        |
READ - CARDS.
        |
        |
        |
    READ CARD - FILE AT END MOVE 'YES' TO END - OF - FILE - FLD.
```

(*) عبارة N = 10 أو عبارة IS EQUAL مستخدمة في لغة الكوبول.

٢ ـ ٣ ـ ٩ تعليمة المقارنة IF

ان تعليمة المقارنة IF هي التعليمة الأكثر استخداماً في برامج الكوبول وذات أثر فعال وكبير في تحويل مسار البرنامج وتأخذ التعليمة الشكل العام التالي :

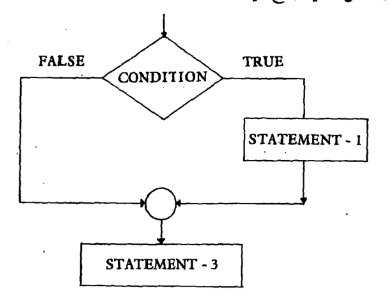
```
                ┌ Statement - 1      ┐   ┌ ┌ ELSE Statement - 2      ┐ ┐
IF Condition    ┤                    ├   │ ┤                         ├ │
                └ NEXT SENTENCE      ┘   └ └ ELSE NEXT SENTENCE      ┘ ┘
```

حيث أن كلمة أو عبارة Condition يقصد بها المقارنة المنطقية بين شيئين Relational Condition ويمكن أن تأخذ أحد الأشكال التالية :

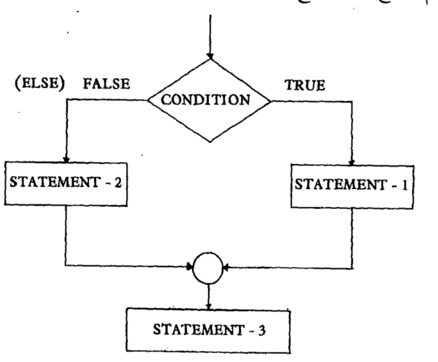
```
                        ┌ IS [NOT] LESS THAN      ┐
┌ identifier - 1   ┐    │ IS [NOT] <              │    ┌ identifier - 2   ┐
┤ literal - 1      ├    ┤ IS [NOT] EQUAL TO       ├    ┤ literal - 2      ├
└ index - name - 1 ┘    │ IS [NOT] =              │    └ index - name - 2 ┘
                        │ IS [NOT] GREATER THAN   │
                        └ IS [NOT] >              ┘
```

وتعليمة IF يمكن أن تحتوي على عبارة ELSE الاختيارية وعند تحقق الشرط الموصوف بعد كلمة IF ينفذ الحاسب الجزء التالي وهو Statement - 1 وإذا لم يتحقق الشرط ينفذ الحاسب التعليمة التالية لتعليمة IF (أو بالأصح الجملة Sentence التالية لجملة IF لأن IF يجب أن تنتهي بنقطة في آخرها) .

والشكل التالي يوضح سير عملية المقارنة واتجاه تنفيذ التعليمة .



رسم لتوضيح تعليمة IF مع عبارة ELSE :



أمثلة :

```
1)   IF FLD - 1 IS EQUAL TO 99 GO TO A - END.
2)   IF VALUE - FLD < 1000 PERFORM MOVE - FLDS
                  THRU E - MOVE
           ELSE DISPLAY "ERROR INPUT"
                  LINE 21 POSITION 10,
                  ACCEPT WAIT - FLD.
3)   IF SUB - CO - E = "1" MOVE SUB - NAME TO FLD - 1 ELSE
                  MOVE SUB - NAME TO ERR - FLD.
```

ويجب الإشارة هنا الى أن المقارنة يمكن أن تكون على حقول رقمية أو حقول غير رقمية، ومن حيث المبدأ فان أية مقارنة تتم رقمياً بوساطة قيمة الأحرف والإشارة الممثلة في الحقل، إلا أنه على المبرمج ملاحظة المقارنة غير الرقمية تتم ابتداء من جهة الشمال في الحقلين محل المقارنة، فمثلا كلمة BAKER مع كلمة BROWN فنجد BAKER أصغر من BROWN لأن قيمة الحرف B و R أكبر من B و A . و ينصح في مثل هذه المقارنة الرجوع الى مراجع الشركة الصانعة للحاسب للتأكد من قيمة مجموعة الاحرف والاشارات المستخدمة.

أما المقارنة الرقمية فتتم على أساس جبري أي العدد السالب أصغر من الموجب.

٢ - ٤ التعليمات الحسابية ARITHMETIC VERBS OR INSTRUCTIONS

في هذا الجزء من لغة كوبول سنستعرض استخدام التعليمات الحسابية في البرنامج وصيغها المختلفة.

٢ ـ ٤ ـ ١ تعليمة الجمع ADD

تستخدم تعليمة ADD لأجل انجاز عملية الجمع بأشكالها المختلفة (أي الجمع الى مخزن واحد وتراكم القيمة أو الجمع الى مخزن آخر) وتأخذ تعليمة الجمع شكلين أساسيين* وهما :

```
                ┌ identifier - 1 ┐  ┌ identifier - 2 ┐
1)   ADD       {                  }  │                │ ... TO  identifier - n
                └ literal - 1    ┘  └ literal - 2    ┘

               [ identifier - m ]

                ┌ identifier - 1 ┐  ┌ identifier - 2 ┐  ┌ identifier - 3 ┐
2)   ADD       {                  }  │                │  │                │
                └ literal - 1    ┘  └ literal - 2    ┘  └ literal - 3    ┘

               .... GIVING   identifier - m   [ identifier - n ]
```

ففي الشكل الأول يتم جمع قيم المتغيرات (أو الثوابت) 1 ,2 ... باضافة النتيجة الى المتغير m أو n الوارد بعد كلمة TO . (أي أن قيمة 1 ,2 ,3, ... ,n تخزن في نفس n)

بينما الشكل الثاني الذي يحتوي على كلمة GIVING فان عملية الجمع تتم باضافة القيم الواردة في المتغيرات (أو الثوابت) 1 , 2 , 3 ,... ووضع النتيجة في المتغير m أو n الوارد بعد كلمة GIVING دون الأخذ بعين الاعتبار القيمة الموجودة في هذه المتغيرات الأخيرة m , n قبل تنفيذ الجمع.

(*) أغلب الحاسبات تحتوي مترجم لا يقبل الجمع أو الطرح أو الضرب أو القسمة ووضع النتيجة في أكثر من مخزن واحد للنتيجة، لذا يراعى الاطلاع على المترجم المستخدم.

مثال :

```
ADD A B TO C
```

قبل التنفيذ	C	15	B	13	A	7
بعد التنفيذ	C	35	B	13	A	7

فالنتيجة 35 تمثل قيمة جمع المتغير A والمتغير B والمتغير C ووضع القيمة النهائية في المتغير C.

مثال :

```
ADD A B GIVING C.
```

قبل التنفيذ	C	15	B	13	A	7
بعد التنفيذ	C	2ϕ	B	13	A	7

وهنا تستبدل القيمة الأولية للمتغير C بنتيجة (أو حاصل) جمع المتغير A و B والتي تعادل 2ϕ، والجدول التالي يلخص تعليمة الجمع في أمثلة :

تعليمة الجمع ADD

Data name :	A	B	C	اسم الحقل :
Value before execution :	15	2ϕ	25	قيمة الحقل قبل التنفيذ :
Value After execution of :				القيمة بعد تنفيذ أي من
ADD A TO C.	15	2ϕ	4ϕ	العمليات الواردة في التعليمات :
ADD A B TO C	15	2ϕ	6ϕ	
ADD A 13 B GIVING C.	15	2ϕ	48	
ADD A 13 B TO C.	15	2ϕ	73	
ADD 1 TO B, C.	15	21	26	

٢ ـ ٤ ـ ٢ تعليمة الطرح SUBTRACT

تستخدم تعليمة الطرح SUBTRACT لأجل حساب تفاضل قيمتين أو أكثر وحساب الفرق بينهما . وتأخذ الشكلين الأساسيين العامين :

```
1)   SUBTRACT  { identifier - 1 }  [ identifier - 2 ] .....
               { literal - 1    }  [ literal - 2    ]

          FROM identifier - m     identifier - n    ......


2)   SUBTRACT  { identifier - 1 }  [ identifier - 2 ] .....
               { literal - 1    }  [ literal - 2    ]

     FROM { identifier - m }    GIVING   identifier - n
          { literal - m    }

          [ identifier - O ] ....
```

ففي الشكل (١) فان قيمة المتغير (الحقل) m تستبدل بقيمة حاصل الطرح (نتيجة الطرح = الحقل m ناقصاً الحقل ١ و ٢ ....).

أما في الشكل (٢) فان قيمة الحقل (m) أو الثابت (m) لا تتغير في عملية الطرح إنما يخزن حاصل الطرح بين القيمة المعرفة في الحقل m أو الثابت m وبقية الحقول أو الثوابت (١) و (٢) و ... في الحقل (n) والحقل (o) .

مثال ١ : SUBTRACT A FROM B.

قبل التنفيذ	B	15	A	1ϕ
بعد التنفيذ	B	5	A	1ϕ

في المثال (١) فان فعل الطرح أدى إلى تخزين حاصل عملية التفاضل بين الحقل B والحقل A (حيث تتم عملية طرح A من B) و يلاحظ أن قيمة A لم تتغير بعد تنفيذ عملية الطرح.

مثال ٢ : SUBTRACT A FROM B GIVING C.

قبل التنفيذ	C	2ϕ	B	15	A	1ϕ
بعد التنفيذ	C	5	B	15	A	1ϕ

يلاحظ في المثال (٢) مع استعمال عبارة GIVING أن قيمة الحقل A قد طرحت من الحقل B ووضعت نتيجة الطرح في المخزن C بغض النظر عن القيمة الموجودة فيه في الأصل. كما أنه يلاحظ عدم تبدل قيمة A و B بعد تنفيذ عملية الطرح. والجدول التالي يوضح أمثلة أوسع عن تعليمة الطرح.

SUBTRACT تعليمة الطرح

Data name :	A	B	C	D	اسم الحقل :
Value before execution :	5	2ϕ	25	3ϕ	قيمة الحقل قبل التنفيذ :
Value after execution of :					قيمة الحقل بعد التنفيذ :
SUBTRACT A FROM C.	5	2ϕ	2ϕ	3ϕ	
SUBTRACT A B FROM C.	5	2ϕ	ϕ	3ϕ	
SUBTRACT A B FROM C GIVING D.	5	2ϕ	25	ϕ	
SUBTRACT 10 FROM C, D.	5	2ϕ	15	2ϕ	

٢ - ٤ - ٣ تعليمة الضرب MULTIPLY

تأخذ تعليمة الضرب في لغة كوبول الشكلين الأساسيين العامين التاليين :

```
                  ┌ identifier - 1 ┐
1)   MULTIPLY     {                }  BY identifier - 2   [ identifier - 3 ]
                  └ literal - 1    ┘

                  { identifier - 1 }       { identifier - 2 }
2)   MULTIPLY     {                }  BY   {                }   GIVING
                  { literal - 1    }       { literal - 2    }

                  identifier - 3   [ identifier - 4 ] ....
```

وتعليمة الضرب تقوم بضرب الحقل (١) بالحقل (٢) وتخزين الجداء في الحقل (٢) و (٣). وذلك مع عدم استعمال عبارة GIVING. بينما في حال استخدم عبارة GIVING فإن نتيجة الضرب (جداء الحقل (١) و (٢)) تخزين في الحقل (٣) و (٤).

مثال ١ : MULTIPLY A BY B.

قبل التنفيذ :	A	5	B	1ϕ
بعد التنفيذ :	A	5	B	5ϕ

و يلاحظ في هذا المثال أن جداء الحقل A و B قد وضع في الحقل B وتبدلت قيمته من 1ϕ الى 5ϕ بينما الحقل A لم يتبدل.

مثال ٢ :

MULTIPLY A BY B GIVING C.

قبل التنفيذ :   A : [5]   B : [10]   C : [15]

بعد التنفيذ :   A : [5]   B : [10]   C : [50]

يلاحظ أن قيمة الحقل C قد استبدلت بالقيمة 50 وهي جداء عملية الضرب A في B، وأن قيمة الحقلين A و B بعد التنفيذ لم تتبدل. والجدول التالي يوضح أمثلة أخرى عن تعليمة الضرب.

MULTIPLY تعليمة الضرب

Data name :	A	B	C	اسم الحقل :
Value Before execution :	5	10	20	قيمة الحقل قبل التنفيذ :
Value after execution of :				قيمة الحقل بعد التنفيذ :
MULTIPLY B BY A GIVING C.	5	10	50	
MULTIPLY A BY B GIVING C.	5	10	50	
MULTIPLY A BY B	5	50	20	
MULTIPLY B BY A.	50	10	20	
MULTIPLY A BY 3 GIVING B, C.	5	15	15	

٢ ـ ٤ ـ ٤ تعليمة التقسيم DIVIDE

عملية التقسيم تنفذ في لغة كوبول بوساطة الفعل DIVIDE وتأخذ تعليمة التقسيم عندئذ الاشكال الأساسية التالية :

1) <u>DIVIDE</u> { identifier - 1 | literal - 1 } <u>INTO</u> identifier - 2 [identifier - 3]

2) DIVIDE { identifier - 1 / literal - 1 } { INTO / BY } { identifier - 2 / literal - 2 } GIVING

identifier - 3 [identifier - 4]

3) DIVIDE { identifier - 1 / literal - 1 } { INTO / BY } { identifier - 2 / literal - 2 } GIVING

identifier - 3 REMAINDER identifier - 4

ففي الشكل (١) يحفظ حاصل القسمة (قسمة الحقل (٢) على الحقل (١)) في الحقل (٢) وتلغى القيمة الأولية للحقل. وإذا كان هناك أكثر من حقل أي (٣) أو (٤) فان حاصل قسمة هذه الحقول (المقسومة) تحفظ في نفس مكان القيمة الأولية (أي الرقم المقسوم) على التتالي.

أما في الشكل (٢) فان حاصل قسمة الحقل (أو العدد الثابت) (٢) على الحقل (١) باستخدام كلمة INTO أو حاصل قسمة الحقل (١) على الحقل (٢) باستخدام كلمة BY يحفظ في الحقل (٣) بدلاً من القيمة الأولية التي كانت فيه، وإذا كان هناك حقل رقم (٣) فان حاصل القسمة يحفظ فيه أيضاً.

أما في الشكل (٣) فإن حاصل القسمة يحفظ في الحقل (٣) بينما يحفظ الباقي من عملية التقسيم REMAINDER في الحقل (٤).

مثال ١ : DIVIDE A INTO B

قبل التنفيذ : A : 2∅ B 1∅∅

بعد التنفيذ : A : 2∅ B 5

مثال ٢ : DIVIDE B BY A GIVING C.

أو DIVIDE A INTO B GIVING C.

قبل التنفيذ :	A :	2ϕ	B	1ϕϕ	C :	13
بعد التنفيذ :	A :	2ϕ	B	1ϕϕ	C :	5

يلاحظ في المثال (١) أن قيمة الحقل B بعد التقسيم تأخذ حاصل قسمة العدد 1ϕϕ على 2ϕ وهو الرقم 5. أما في المثال الثاني فان حاصل قسمة الحقل B على A يوضع في الحقل C (أي 5) و بغض النظر عن القيمة الأ ولية الموجودة أصلاً فيه وهي (13).

مثال ٣ : DIVIDE A INTO B GIVING C

REMAINDER D.

قبل التنفيذ :	A :	2ϕ	B	1ϕ2	C :	13	D :	15
بعد التنفيذ :	A :	2ϕ	B	1ϕ2	C :	5	D :	2

يلاحظ في هذا المثال أن حاصل قسمة الحقل B على A قد خزنت في الحقل ٢ عوضاً عن القيمة (13) بينما تم حفظ الباقي من عملية التقسيم وهو العدد 2 في الحقل D بدلا من القيمة 15. و يلاحظ أيضاً أن قيمة الحقلين A و B لم تتبدل بعد تنفيذ عملية التقسيم.

والجدول التالي يوضح أمثلة مختلفة عن تعليمة التقسيم.

تعليمة التقسيم DIVIDE

Data name :	A	B	C	اسم الحقل :
Value before execution :	1ϕ	2ϕ	6ϕ	قيمة الحقل الأ ولية :

Value after execution of : قيمة الحقل بعد التنفيذ :

DIVIDE 4 INTO B.	1ϕ	5	6ϕ
DIVIDE 4 INTO B GIVING C.	1ϕ	2ϕ	ϕ5
DIVIDE B BY 10 GIVING A.	ϕ2	2ϕ	6ϕ
DIVIDE A INTO B, C.	1ϕ	ϕ2	ϕ6
DIVIDE A INTO B GIVING C.	1ϕ	2ϕ	ϕ2
DIVIDE 3 INTO A GIVING B REMAINDER C.	1ϕ	3	1

٢ ـ ٤ ـ ٥ تعليمة احسب COMPUTE

تعليمة COMPUTE يمكن أن تنفذ أي عملية حسابية سابقة مثل الجمع، الطرح، الضرب، أو التقسيم بالاضافة الى امكانية احتوائها على عمليات مختلفة (أو مركبة) أي جمع مع طرح مع ضرب أو مع تقسيم. فمثلا التعبير الحسابي التالي :

Y = 5 (A + B) / C

وفيه قيمة المتغير Y عبارة عن عمليات جمع وضرب وتقسيم، فلو أردنا كتابته بالتعليمات السابقة فيكون :

```
ADD A  B GIVING Y.
MULTIPLY 5 BY Y.
DIVIDE C INTO Y.
```

بينما عند استخدام تعليمة COMPUTE فيمكن كتابة هذه العمليات الحسابية على النحو التالي :

```
COMPUTE Y = 5 * (A + B) / C.
```

و يكون الشكل العام لتعليمة COMPUTE :

<u>COMPUTE</u> identifier - 1 = arithmetic expression

حيث أن التعبير الرياضي (أو الحسابي) Arithmetic expression يجب أن يصاغ

على النحو التالي :

١ - ان الاشارات + ، - ، * ، / ، ** تشير الى الجمع ، الطرح ، الضرب ، التقسيم والرفع الى قوة على التتالي .

٢ - يمكن أن يحتوي على أسماء حقول أو ثوابت literal أو اشارات حسابية وأقواس . ويجب أن يكون هناك فراغ قبل الاشارة الحسابية وفراغ بعدها .

٣ - تستخدم الأقواس ( ) لتوضيح تنفيذ العمليات الحسابية المتسلسلة في نفس التعبير الحسابي . ويجب أن يكون أي تعبير حسابي محصور بين قوسين موافق للشروط العامة للتعبير الحسابي في تعليمة COMPUTE . و يسبق القوس الأ ول فراغ ويجب أن يلي القوس الآخر فراغ أيضاً .

وتقوم تعليمة COMPUTE بحساب القيمة المبينة في التعبير الحسابي على يمين إشارة المساواة '=' ثم تخزن نتيجة العمليات الحسابية المختلفة في الحقل (أو المتغير) المبين الى يسار اشارة المساواة . ويجري تقييم الصيغة الحسابية على النحو التالي :

١ - يتم حساب قيمة الصيغة داخل الأقواس أولا و بشكل منفصل .

٢ - وضمن الصيغة الحسابية الواحدة يتم تنفيذ العمليات الحسابية من جهة اليسار الى اليمين وفق الترتيب التالي :

- الرفع الى قوة
- الضرب والتقسيم
- الجمع والطرح

٣ - وعند وجود عمليتين حسابيتين من نفس المستوى ، مثلا : جمع وطرح يتم تنفيذهما اعتباراً من جهة اليسار.

وتجدر الاشارة هنا الى أن بعض أنواع الحاسبات لا تسمح باستخدام تعليمة COMPUTE لأنها تعتبرها أقرب الى اللغات العلمية منها الى اللغة التجارية كوبول فيجب الرجوع الى مراجع الشركة الصانعة للحاسب عند تطبيق هذه التعليمة .

والجدول التالي يوضح أمثلة عن تعليمة COMPUTE :

Data name :	A	B	C	اسم الحقل :
Value before execution :	4	6	2ϕ	القيمة الأولية للحقل :
				قيمة الحقل قبل تنفيذ
Value After execution of :				أي من التعليمات :
COMPUTE C = A + B.	4	6	1ϕ	جمع بسيط
COMPUTE C = A + B * 5	4	6	34	الضرب يتم قبل الجمع
COMPUTE C = (A + B) * 5	4	6	5ϕ	حساب الأقواس يتم قبل
COMPUTE C = A ** 2.	4	6	16	تعادل جبرياً $c = a^2$
COMPUTE C = B ** A.	4	6	1296	تعادل جبرياً $c = b^a$

٢ ـ ٥ اسئلة وتمرينات

سؤال ١ ـ ما هي الأخطاء التي يتوقع وجودها مترجم لغة كوبول عند ترجمته لهذا الجزء من البرنامج ؟

```
DATA DIVISION.
FILE - SECTION.
        FD IN - FILE.
        LABEL RECORD IS STANDARD.
        ϕ1      A - REC.
        ϕ3      A - 1      PIC     x (6).
        ϕ3      A - 2.
        ϕ5      A - 3      PIC     x (8).
        ϕ7      A - 4      PIC     999.
```

```
ϕ7    A - 5    PIC    xxxx.
ϕ7    A - 6.
ϕ5    A - 7    PIC    x (2ϕ)
ϕ2    A - 8    PIC    9    9.
```

سؤال ٢ - اذا كان لدينا ملفان على بطاقات مثقبة الأول اسمه A - FILE والآخر اسمه B - FILE. وهناك ملف ثالث محجوز على القرص الممغنط اسمه C - FILE لأجل كتابة السجلات عليه.

هذه الملفات تحتوي السجلات على التتالي A - REC, B - REC, C - REC وقسم الاجراءات في برنامج كوبول مصمم لانجاز ما يلي :

أ - فتح الملفات

ب - قراءة الملف A - FILE

جـ - قراءة الملف B - FILE وكتابة سجل على الملف C - FILE

د - تخزين (أو حفظ) السجل A - REC في السجل C - REC

هـ - في نهاية الملف A - FILE يجب اعادة فتحه وقراءته من أوله واعادة المعالجة من الخطوة رقم (ب)

و - في نهاية الملف B - FILE تغلق الملفات و ينتهي البرنامج.

والمطلوب تدقيق هذا الجزء من برنامج كوبول الذي من المفترض أن ينجز هذه الاجراءات :

```
PROCEDURE DIVISION.
        OPEN INPUT A - FILE OUTPUT B - FILE.
```

```
A - READ.
        READ A - FILE AT END GO TO A - END.
        READ B - FILE AT END CLOSE B - FILE STOP RUN.
        OPEN OUTPUT C - FILE.
        WRITE C - FILE FROM B - FILE.
        MOVE A - REC TO C - REC.
        GO TO A - READ.
A - END.
        CLOSE A - FILE.
        MOVE A - REC TO C - REC.
        OPEN INPUT A - FILE.
        GO TO A - READ.
```

سؤال ٣ ـ أكتب التعليمات التي تقوم بما يلي :

أ ـ ملء المخزن REC - COUNTS بالأصفار

ب ـ إعطاء القيمة 19 للمخزن المسمى A - QTY

جـ ـ إعطاء القيمة 1 للمخزن المسمى B - QTY

د ـ وضع العبارة COBOL EXERCISE في المخزن المسمى A - AREA

سؤال ٤ ـ أكمل الجدول التالي للتعليمات الصحيحة فقط، حيث أن قيم الحقول A, B, C, D معطاة كقيمة أولية قبل تنفيذ التعليمة، وبين سبب عدم صحة التعليمة الخطأ

Field - name :	A	B	C	D
Initial - value :	4	5	6	7
MOVE C TO SPACE.	_	_	_	_
MOVE ZERO TO A, B, C.	_	_	_	_
MOVE SPACE TO A, B.	_	_	_	_
MOVE D TO B.	_	_	_	_
MOVE 'X' TO A.				

الفصل الثالث

الأقسام الأخرى في برنامج كوبول

٣ ـ ١ لمحة عامة

علمنا من الفصل الأول أن برنامج كوبول يتألف من أربعة أقسام رئيسية وهي قسم التعريف وقسم محيط البرنامج وقسم البيانات وقسم الاجراء وبينا في الفصل الثاني التعليمات المهمة الأولية التي يمكن أن ترد في قسم الاجراء. أما في هذا الفصل فسوف ننظر في كيفية كتابة الأقسام الأخرى وبشكل خاص قسم البيانات DATA DIVISION لأنه القسم الأكثر أهمية بالنسبة لباقي الأقسام. كما سوف يتم في هذا الفصل شرح أرقام المستوى وعبارة الصورة Picture ثم الفصل الخاص بتوصيف البيانات في التخزين الداخلي للبرنامج ثم أشكال اخراج البيانات وإعطاء الصورة المطلوبة. وبنهاية هذا الفصل يمكن للدارس الإلمام بكل المعلومات الضرورية لكتابة برنامج كوبول أولي يكون نواة لبرنامج أكثر تقدماً في الفصول اللاحقة.

٣ ـ ٢ قسم التعريف IDENTIFICATION DIVISION

يعتبر قسم التعريف أول قسم من الأقسام الأربعة الذي يتألف منها برنامج كوبول ويرد في أول البرنامج. ووظيفة هذا القسم تزويد البرنامج بالمعلومات التعريفية* عن البرنامج نفسه مثل: اسم البرنامج ومؤلف البرنامج، وتاريخ البرنامج

(*) Identifying information

وتاريخ كتابته واسم مركز الحاسب وتاريخ ترجمة البرنامج .. الخ. و يتألف القسم هذا من عنوان أساسي له وأسماء فقرات تالية له يمكن أن تصل الى ست فقرات وفق الشكل التالي :

```
IDENTIFICATION DIVISION.
PROGRAM - ID.       program - name.
[AUTHOR.            Comment - entry.]
[INSTALLATION.      Comment - entry.]
[DATE - WRITTEN.    Comment - entry.]
[DATE - COMPILED.   Comment - entry.]
[SECURITY.          Comment - entry.]
```

و يلاحظ من الشكل أن هناك فقط فقرة واحدة بعد اسم القسم اجبارية وهي PROGRAM - ID وتمثل اسم البرنامج أما باقي الفقرات فهي اختيارية وتدل على التوالي : اسم المؤلف (المبرمج)، اسم مركز الحاسب، تاريخ كتابة البرنامج، تاريخ ترجمة البرنامج، وضمان سلامة البيانات والبرنامج. و يرد في هذه الفقرات بيانات تعتبر بمثابة تعليق وتوثيق للبرنامج لا أكثر من أجل سهولة الرجوع اليه في المستقبل ولذلك دعيت بيانات تعليقية .Comment - entry.

مثال :

```
IDENTIFICATION DIVISION.
PROGRAM - ID.     PROG ϕ1.
AUTHOR.           AUTHORS.
INSTALLATION.     IPA COMPUTER CENTER.
DATE - WRITTEN. FEBRAURY 4, 1983.
```

DATE - COMPILED. FEBRAURY **, 1983.

SECURITY. THERE IS PASSWORD FOR USER.

و يُرمَّز هذا المثال وفق قواعد الكتابة على الورقة الخاصة بلغة كوبول ويجب أن تبدأ جميع أسماء هذه الفقرات من المنطقة A (أي الأعمدة ٨ - ١١). أما محتوى الفقرة فيمكن أن يكتب في المنطقة B (أي الأعمدة ١٢ وما بعد الى ٧٢).

٣ - ٣ قسم البيئة (المحيط) ENVIRONMENT DIVISION

يعتبر قسم محيط البرنامج القسم الثاني في برنامج كوبول من حيث تسلسله في البرنامج ووظيفة هذا القسم تتلخص في نقطتين :

- تعريف الحاسب المستخدم في عملية ترجمة البرنامج وتنفيذه (عادة ما يكون الحاسب نفسه هو المستخدم في عملية الترجمة وعملية التنفيذ). و يظهر هذا التعريف في فصل مكونات الحاسب Configuration section.
- ربط الملفات المستخدمة في البرنامج مع وحدات الإدخال والإخراج المتصلة بالحاسب مباشرة. وهذا يظهر في فصل الإدخال والإخراج ضمن فقرة ضبط الملفات INPUT - OUTPUT SECTION, FILE - CONTROL.

لذلك فقد يختلف هذا القسم في برنامج كوبول باختلاف الحاسب المستخدم و يُنصح بالرجوع الى مراجع الشركة الصانعة للحاسب لتحديد معطيات هذا القسم. و يأخذ فصل تعريف مكونات الحاسب الشكل العام التالي :

```
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE - COMPUTER. Computer - name.
OBJECT - COMPUTER. Computer - name.
```

وتبدأ كتابة أسماء الفقرات في المنطقة A (الأعمدة ٨ - ١١) و بقية المعطيات في المنطقة B (الأعمدة ١٢ - ٧٢).

أما الفصل الثاني الأكثر أهمية في هذا القسم هو فصل الإدخال والإخراج ويحتوي على فقرة تدعى FILE - CONTROL وضمنها يُعرف كل ملف من الملف المستخدمة في البرنامج بعبارة خاصة تدعى SELECT مع كلمة ASSIGN وتعني اختيار وحدة الإدخال والإخراج وتعيينها لاسم الملف المنطقي المستخدم ضمن البرنامج وتقوم هذه العبارة بالربط بين الوحدة المخصصة والملف المستخدم لأجل تعريف البرنامج بجهة مصدر أو مستقبل البيانات من والى البرنامج. وتخصيص أو تعيين وحدات الادخال والاخراج يختلف من حاسب لآخر لذلك ينصح بالعودة الى مراجع الشركة الصانعة للحاسب لهذا الغرض.

والشكل التالي يوضح كيفية هذا الفصل لحاسب شركة IBM :

```
INPUT - OUTPUT SECTION.
FILE - CONTROL.
        SELECT CARD - FILE ASSIGN TO UT - S - SYSIN.*
        SELECT PRINT - FILE ASSIGN TO UT - S - SYSOUT.
```

و يقصد بعبارتي SELECT ما يلي : تخصيص وحدة الادخال المسماة UT - S - SYSIN لملف الادخال المسمى في البرنامج باسم CARD - FILE وكذلك تخصيص وحدة الاخراج المسماة في نظام التشغيل UT - S - SYSOUT للملف الاخراج المسمى في البرنامج باسم PRINT - FILE.

(*) حسب توصيف IBM 3033 ونظام التشغيل OS/MVS.

أما اذا أردنا كتابة نفس الفصل لحاسب TI 99ɸ فإنه يأخذ الشكل التالي :

```
INPUT - OUTPUT SELECT.
FILE - CONTROL.
        SELECT CARD - ASSIGN TO INPUT "SYN"*
        SELECT PRINT - FILE ASSIGN TO PRINT "SYN"
```

وكما سبق فان اسماء الفقرات تكتب ابتداء من المنطقة A (الأعمدة ٨ - ١١) وعبارة SELECT تبدأ من المنطقة B (الأعمدة ١٢ - ٧٢).

٣ - ٤ قسم البيانات DATA DIVISION

يأتي قسم البيانات في برنامج كوبول بعد قسم محيط البرنامج و يصف هذا القسم جميع الحقول وأماكن التخزين في الذاكرة اللازمة للبرنامج. كما يحتوي في وصفه على حجم كل حقل بالرموز Characters وكذلك نوعية هذا الحقل مثل رقمي أو أبجدي أو غير رقمي. و يوضح أيضا هيكل أو تركيب البيانات المستخدمة. والفقرة التالية تبين كيفية وصف هذه المعطيات بوساطة عبارة الصورة Picture وأرقام المستو يات المتدرجة.

٣ - ٤ - ١ عبارة PICTURE

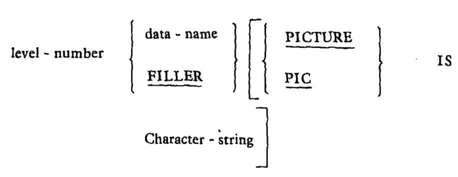
في قسم البيانات يجب وصف البيانات وتحديد أسمائها وكذلك حجم البيانات ونوعيتها و يتم ذلك بوساطة عبارة الصورة PICTURE. ومبدئيا هناك ثلاثة أنواع مهمة من البيانات وهي : الأبجدية و يرمز لها بالحرف A والرقمية بالرقم 9. والحرفية - الرقمية** بالرمز X. أما حجم البيانات (أي طول الحقل المستخدم، فيحسب بعدد

(*) حسب توصيف TI 990 نظام التشغيل D × 12 وكلمة SYN يقصد بها اسم الملف الرمزي والذي يجب أن يعرف للحاسب (نظام التشغيل) بأمر COMMAND يدعى AS (ASSIGN SYNONYM).

(**) Alpharnumeric

الرموز (أحيانا يكون كل رمز مساو ياً لبايت Byte واحدة) . و يعبر عن ذلك بتكرار رمز نوع البيانات بعدد المرات مساو ياً لطول الحقل المرغوب. فمثلاً إذا كان الحقل أبجدياً ومساو ياً لأ ربع رموز فيكتب على شكل AAAA أو (4) A ، أما اذا كان الحقل رقمياً وطوله ثلا ثة أرقام فيكتب وصفه على الشكل 999 أو (3) 9 أما اذا كان الحقل غير رقمي وكان طوله فرضا ستة رموز فيكتب (6) X أو XXXXXX .

وتأخذ هذه العبارة الشكل العام التالي الأ ولي :



حيث يبيّن رقم المستوى level - number تسلسل الحقل الموصوف و data - name اسم المتغير أو الحقل الموصوف و character - string يمثل نوع وحجم الحقل الموصوف . أما الكلمات FILLER PICTURE, PIC, IS فهي كلمات كوبول محجوزة و يقصد بكلمة FILLER اسم لحقل لا يستخدم من قبل المبرمج ويمثل حقل DUMMY ضمن البيانات .

٣ ـ ٤ ـ ٢ أرقام المستويات LEVEL NUMBERS

تصنف حقول البيانات data - items في لغة كوبول الى نوعين : حقول اجمالية group وحقول ابتدائية (أولية) elementary . والحقل الاجمالي عبارة عن مجموعة حقول ابتدائية أما الحقل الابتدائي فهو الحقل الذي لا يمكن تجزئته الى حقول أخرى . والشكل التالي يبين مثالا على هذا التصنيف :

STD - CARD					
STD - NO	STD - NAME		MARKS		SPACE (NOT USED)
	STD - LAST-NAME	STD - FIRST - NAME	SCORE	GRADE	

1 7 8 22 23 37 38 41 42 43 80

numeric Alphabetic numeric Alpha - numeric غير مستخدم

ففـي هـذا الـشـكـل يـتـضـح هـيـكـل بـيـانـات لـسـجـل (حـقـل اجمالي) يسمى STD - CARD . ومجزأ الى حقول فرعية :

STD - NO , STD - NAME , MARKS

والحقل الجزئي STD - NAME مقسم الى حقلين آخرين

STD - LAST - NAME , STD - FIRST - NAME

وكذلك الحقل MARKS مقسوم الى حقلين :

SCORE, GRADE

فالحقول الاجمالية في هذا الشكل هي STD - CARD

و STD - NAME , MARKS بينما الحقول الابتدائية هي :

STD - NO, STD - LAST - NAME, STD - FIRST - NAME, SCORE, GRADE.

ولاجـل تـوصـيـف هـذا الـتـركـيب من البيانات في لغة كوبول تستخدم أرقام المستوى للـدلالـة على تـدرج تـوزيـع الحقول من اجمالي الى جزئي، والحقل الاجمالي الأول يُعطى الرقـم 01 و يُعطى الحقل أو الحقول المكونة له أرقاماً أكبر مثل 02 وهكذا لغاية الرقم 49 ضـمـنـاً . (كما يوجد هناك الأرقام 66 و 77 و 88 وسوف يرد ذكرها وكيفية استخدامها

في الفصول التالية). ويمكن وضع الشكل السابق من البيانات على صيغة سجل في لغة كوبول على النحو التالي :

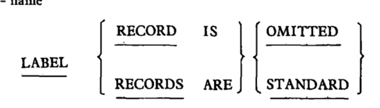
```
ϕ1   STD - CARD.
     ϕ5   STD - NO     PICTURE IS 9 (7).
     05   STD - NAME.
          ϕ7   STD - LAST - NAME    PICTURE IS A (15).
          ϕ7   STD - FIRST - NAME   PICTURE IS A(15).
     ϕ5   MARKS.
          1ϕ   SCORE          PICTURE IS 9 (4).
          1ϕ   GRADE          PICTURE IS X.
     ϕ5   FILLER              PICTURE IS X (38).
```

يلاحظ من هذا التوصيف أن الحقول الابتدائية هي فقط التي تشتمل على عبارة PICTURE كما يلاحظ أيضا أن أرقام المستوى ليس بالضرورة أن تكون متسلسلة . أما الحقل الأخير المسمى FILLER فهويشير الى اسم حقل لجزء من السجل غير مستخدم (أي مغفل في البرنامج) ويجب دائما وصفه بالنوع X لأنه من الجائز أن يكون محتوياً على رموز حرفية ورقمية أو يكون فارغاً . هذا ويجوز استخدام كلمة PIC للدلالة على PICTURE وكلمة IS ليست اجبارية عند كتابة هذه العبارة كما يجب الاشارة هنا إلى أن أرقام المستوى تكتب ابتداء من الأعمدة ٨ ـ ١١ بالنسبة للحقل الأول الاجمالي فقط أما بقية الأرقام فيمكن أن ترد ضمن هذه المنطقة أو بعدها كما أن وجود أكثر من فراغ بين الكلمات في التوصيف ليس خطأ بل يسهل عملية القراءة والمراجعة .

٣ - ٤ - ٣ فصل توصيف الملفات FILE SECTION

يعتبر فصل توصيف الملفات الفصل الأول في قسم البيانات في لغة كوبول وهذا الفصل يشتمل على توصيف لكل ملف تم ذكره في عبارة SELECT في قسم محيط البرنامج وإذا لم يكن هناك ملفات ضرورية في البرنامج المكتوب فلا داعي لذكر هذا الفصل لذلك يمكن اعتباره فصلا اختياريا.

وفصل توصيف الملف يحتوي عادة على وصف للملف يليه توصيف للسجل الخاص بهذا الملف. وقد وصفنا في الصفحات القليلة السابقة سجل لملف طالب STD - CARD ولنفرض أنه من ملف اسمه STD - FILE فلأجل وصف الملف تستخدم الشكل العام الأولي لوصف الملف و يدعى عبارة FD* و يكتب على النحو التالي :



```
FD file - name

         LABEL  { RECORD  IS  } { OMITTED  }
                { RECORDS ARE } { STANDARD }

         [RECORD CONTAINS integer - 1 CHARACTERS]
         [DATA RECORD IS data - name - 1]
```

وعبارة LABEL تصف خلاصة وصف الملف المسجل أو الذي سوف يسجل على وحدات التخزين المغناطيسي في حال كونها STANDARD أما كلمة OMITTED فتستخدم مع ملف الادخال من بطاقات مثقبة أو ملف الاخراج على طابعة أو مع ملف الشريط المغناطيسي أحياناً عندما لا يحتوي على LABEL خاصة به.

(*) تعني File Description

والعبارتان التاليتان الاختياريتان تصفان حجم السجل في هذا الملف واسم السجل الخاص به على التوالي.
وعندئذ يمكن كتابة وصف الملف STD - FILE على النحو التالي :

```
FD      STD - FILE
        LABEL RECORD IS OMITTED*
        RECORD CONTAINS 80 CHARACTERS
        DATA RECORD IS STD - CARD.
```

وحسب قواعد لغة كوبول ترد كلمة FD ابتداء من الأعمدة ٨ ـ ١١ أما بقية العبارات في هذه الجملة فتبدأ بعد العمود ١٢ إلى ٧٢.

٣ ـ ٤ ـ ٤ فصل التخزين الداخلي WORKING - STORAGE SECTION

يستخدم هذا الفصل في قسم البيانات لأجل تخزين النتائج الوسيطة من المعالجة وكذلك تعريف وتوصيف المعطيات الثابتة في البرنامج أو بمعنى آخر يحتوي على توصيف للبيانات التي يحتاجها البرنامج بدون أية قراءة من خارج البرنامج.

وهذا الفصل يحتوي على نوعين للبيانات الموصوفة :

- النوع الأول الحقول (المتغيرات) المستقلة والتي تكون أولية بحد ذاتها ولا تحتاج إلى تجزئة. وهذه الحقول تأخذ رقم المستوى 77 وعادة تذكر في هذا الفصل قبل النوع الآخر**، كما ويمكن أن ترد بعده.

(*) يفترض أنه موجود على بطاقات مثقبة.

(**) إن الاتجاه الحديث في البرمجة يقضي بتجميع الحقول المستقلة التي ليس لها علاقة تركيبة مع غيرها في مجموعة اجمالية من الحقول ثم تأخذ رقم المستوى 01 وبقية الأرقام المتدرجة.

- النوع الثاني الحقول (المتغيرات) الاجمالية والتي يمكن تجزئتها الى حقول جزئية وتأخذ هذه الحقول رقم المستوى ϕ1 والحقول الجزئية تأخذ أرقاماً متدرجة أخرى كما سلف ذكره في الفقرات السابقة.

والشكل التالي يوضح كيفية توصيف البيانات في هذا الفصل :

```
    WORKING - STORAGE SECTION.
77     TOTAL - AREA        PIC 9 (4) VALUE ZEROS.
77     MIN - MARK          PIC 99 VALUE 6ϕ.
ϕ1     HDR - LINE.
       ϕ5    FILLER        PIC X (1ϕ) VALUE SPACES.
       ϕ5    FILLER        PIC X (6) VALUE 'I.P.A.'.
       ϕ5    FILLER        PIC X (1ϕ) VALUE SPACES.
```

٣ ـ ٤ ـ ٥ عبارة تحديد القيمة VALUE

لقد ورد في الفقرة السابقة كلمة VALUE بجانب توصيف الحقل وتعني هذه العبارة تحديد قيمة ثابتة في الحقل أو اعطائه قيمة أولية لا تتبدل أثناء تنفيذ البرنامج إلا بعملية نقل قيمة أخرى على نفس الحقل بوساطة تعليمة MOVE.

وتأخذ عبارة تحديد القيمة الشكل العام التالي :

```
                                  ┌ PICTURE ┐
level - number     data - name    │         │   IS Character - string
                                  └ PIC     ┘

                   [VALUE IS literal]
```

حيث أن كلمة literal تبقى قيمة محددة ويمكن تصنيفها الى ثلاثة أنواع :

١ ـ ثابت رقمي مثل القيمة 60 في الحقل MIN - MARK

٢ ـ ثابت غير رقمي (رقمي / حرفي) و يوضع بين اشارتي ` ´ مثل 'I.P.A.' في الحقل المسمى FILLER

٣ ـ ثابت شكلي (figurative) وهو عبارة عن كلمات محجوزة في لغة كوبول لترمز الى ثوابت شائعة الاستخدام ومقروءة بشكل أفضل. مثل كلمة ZERO (وللجمع تستخدم ZEROS*) في الحقل المسمى TOTAL - AREA وكلمة SPACE (للجمع تستخدم SPACES*) كما في الحقل المسمى FILLER.

وتجدر الإشارة هنا بأنه لا يجوز تحديد قيمة غير رقمية في حقل موصوف بالرمز 9 (رقمي) مثل :

```
ϕ5   A   PIC   9   VALUE Y.        (غير صحيح)
```

كما يجب دائما عند تحديد القيمة غير الرقمية في الحقل غير الرقمي وضع اشارتي ` ´ (الفاصلة العلوية) لتحديد بداية ونهاية القيمة الثابتة. مثل :

```
ϕ5   B   PIC   X   VALUE 8.        (غير صحيح)
```

وكأمثلة صحيحة نذكر :

```
77   C   PIC   9     VALUE 9.
ϕ1   D   PIC   XXX   VALUE 'IPA'.
```

(*) الكلمات المفردة والجمع متعادلة في لغة كوبول أي SPACE تعادل ولها نفس الوظيفة مثل SPACES بل ذكرت بالجمع لأجل القراءة الواضحة باللغة الانكليزية.

٣ ـ ٤ ـ ٦ الفاصلة العشرية المفترضة ASSUMED DECIMAL POINT

من أجل توفير المساحة في وساطات ادخال البيانات (مثل البطاقة المثقبة سابقاً أو الوسائط المغنطيسية) لا يسمح بادخال الفاصلة العشرية ضمن رموز الرقم أو الحقل المراد قراءته الى البرنامج وإنما يفترض وجود هذه الفاصلة العشرية في الرقم، و يستطيع الحاسب (من خلال البرنامج) تمييز الجزء العشري والجزء الصحيح من الرقم عن طريق وصف صورة الحقل PICTURE التي تحتوي إشارة معينة وهي الرمز V للدلالة على مكان وجود هذه الفاصلة العشرية في الرقم .. فمثلاً الرقم 12345 يقرأ على أنه 123.45 إذا كان موصوفاً في قسم البيانات بعبارة PICTURE على النحو التالي :

```
:
05 AMOUNT PIC IS 999V99.
:
```

ولا يأتي الرمز في عبارة «الصورة» إلا مع النوع الرقمي من البيانات وهو الرمز 9. ويجب التأكيد هنا أن الرمز V يعني وجود فاصلة عشرية في الرقم الموصوف ولكنه لا يُعد أو يحسب ضمن حجم الرقم أو الحقل. فمثلاً الحقل المذكور أعلاه AMOUNT يكون حجمه خمسة أماكن (بايت) فقط ضمن الذاكرة.

ولزيادة الإيضاح : إذا كان لدينا بطاقة مثقبة تحتوي على الأرقام التالية والمتصلة مع بعضها :

```
1                                   80
1 2 3 4 5 6 7 8 9 0 .....
```

وكانت هذه البطاقة موصوفة في قسم البيانات على النحو التالي :

```
ϕ1   PUNCH - CARD.
     ϕ5   FLD - A     PIC   9V99.
     ϕ5   FLD - B     PIC   99V9.
     ϕ5   FLD - C     PIC   9.
     ϕ5   FLD - D     P2C   V999.
     ϕ5   FILLER      PIC   X(7ϕ).
```

فتكون قيمة الحقول : FLD - A, FLD - B, FLD - C, FLD - D
كما يلي على التوالي :

1.23 , 45.6 , 7 , .890

٣ ـ ٤ ـ ٧ اظهار البيانات الرقمية EDITING NUMERIC DATA

عادة لا تحتوي البيانات الرقمية المدخلة سوى أرقام فقط (digits) ولكن بالمقابل لا بد من قراءة البيانات المخرجة والتي يجب أن تشتمل على الرموز المساعدة في فهم الأرقام ومعناها . لذلك تتم عملية ادخال الفاصلة العشرية والفاصلة العادية ورمز العملة على الحقل الرقمي عند إخراجه على الطابعة و يسمى الحقل الذي يظهر الرقم بهذا الشكل المقروء بحقل الاظهار (Edited Field) .
وهذه العملية توفر مساحة في الذاكرة لعدم حجز أماكن لهذه الرموز وتعطي امكانية قراءة هذه الأرقام بعد اظهارها وتقديمها للمستفيد من البرنامج .

ولنأخذ في هذا القسم فقط ثلاثة رموز من رموز الاظهار وهي كافية في المراحل الأولية من لغة كوبول . وهذه الرموز الفاصلة العشرية ، والفاصلة العادية ورمز العملة .

ولنبدأ **بالفاصلة العشرية** : فإذا كان لدينا الحقل الرقمي المسمى FLD - A والموصوف على النحو التالي مع حقل الاظهار والخاص به والمسمى EDT - FLD - A :

```
ϕ5    FLD - A    PIC    9V99.
ϕ5    EDT - FLD - A    PIC    9.99.
```

و يلاحظ وجود فاصلة عشرية (Decimal Point) [في نفس مكان الرمز V الذي يدل على وجود الفاصلة في الرقم ضمنا] في حقل الاظهار وهذه الفاصلة العشرية تحسب من حجم الحقل أي أنه يتكون من أربعة أماكن (بايت) يتم حجزها على وسيط الاظهار (الطابعة أو الشاشة) و باستخدام تعليمة النقل MOVE يكون (بفرض قيمة الحقل 567) .

	EDT - FLD - A	FLD - A
قبل تنفيذ النقل :	? . ? ?	5 V 6 7
بعد تنفيذ النقل : MOVE	5 . 6 7	5 V 6 7

أما **الفاصلة العادية** : التي تُحشر بين الأرقام للدلالة على قيمة الألوف في الحقل فإنها أيضاً توصف ضمن حقل الاظهار وتحسب عند حساب أماكن (مواضع ، بايت) التي يحجزها الحقل في الذاكرة أو على الطابعة .

وكمثال على ذلك نأخذ الحقل FLD - F الذي نريد إظهاره وفق الحقل EDT - FLD - F والموصوفين كما يلي في قسم البيانات :

```
ϕ5    FLD - F         PIC 9 (5).
ϕ5    EDT - FLD - F         PIC 99,999.
```

وباستخدام تعليمة النقل MOVE يكون (على فرض أن قيمة الحقل 12345) ما يلي :

	EDT - FLD - F	FLD - F
قبل تنفيذ النقل	? ? , ? ? ?	1 2 3 4 5
بعد تنفيذ النقل	1 2 , 3 4 5	1 2 3 4 5

أما اشارة العملة المستعملة* : (مثلا الدولار $ وهي الموجودة في اغلب نسخ لغة كوبول) فهي تُوصف في حقل الاظهار وعلى الجهة اليسرى منه. كما يمكن أن تجتمع رموز الاظهار معاً في توصيف الحقل الرقمي المراد اخراجه ولنأخذ مثلاً الحقل EDT - FLD - M الذي يمثل حقل اظهار للحقل FLD - M والموصوفين في قسم البيانات على النحو التالي :

```
ϕ5   FLD - M          PIC 9 (4) V99.
ϕ5   EDT - FLD - M    PIC $ 9,999.99.
```

وباستخدام تعليمة النقل MOVE يكون (بفرض القيمة تساوي 123456)

قبل تنفيذ النقل :

EDT - FLD - M	FLD - M
$? , ? ? ? . ? ?	1 2 3 4 5 6

بعد تنفيذ النقل :

EDT - FLD - M	FLD - M
$ 1 , 2 3 4 . 5 6	1 2 3 4 5 6

(*) يمكن استخدام رموز عملة أخرى على شرط أن تكون متوفرة على وسيط الاظهار (الطابعة أو الشاشة).

٣ ـ ٥ برنامج كوبول متكامل :

يوجد في الصفحات التالية صورة عن برنامج كوبول متكامل و يقوم هذا البرنامج بقراءة بيانات تحتوي على الاسم والراتب ومكان الاقامة والعمر ثم طباعة البيانات للأشخاص والذين تتحقق فيهم الشروط التالية :

الراتب : أن يكون بين ٢٠٠٠ و ٣٠٠٠ ريال

مكـــان : الاقامة أن يكون مدينة الرياض RIYADH

العمـــر : أن يكون أكبر من ٣٦ عاماً .

```
*       برنامج كوبول متكامل
*     ============================
*
*
* : قســم التعريف بالبرنامج
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT23.
 AUTHOR.      KHAYAT-JNAID.
*
*  يرى هذا التمرين الخطوات الاساسيه فى كتابه
*   برنامج كوبـــــــــــــــــــــــــــول
*
*: قســـــم المحيطيات
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.    IBM-370.
   OBJECT-COMPUTER.    IBM-370.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
       SELECT IN-FILE ASSIGN  TO UT-S-INF.
       SELECT P-FILE  ASSIGN  TO UT-S-PRFL.
*
*: قســـــم توصيف البيانات
 DATA DIVISION.
 FILE  SECTION.
```

```
*     : توصيف ملف الإدخال
*
 FD     IN-FILE   LABEL RECORD  IS STANDARD
                        DATA RECORD IS IN-REC.

 01     IN-REC.
        03    FILLER    PIC  X(05).
        03    NAME      PIC  A(15).
        03    FILLER    PIC  X(04).
        03    SALARY    PIC  9(04).
        03    FILLER    PIC  X(11).
        03    LOCATION  PIC  X(11).
        03    AGE       PIC  9(2).
        03    FILLER    PIC  X(28).

*
*     : توصيف ملف الطباعة
*
 FD     P-FILE    LABEL RECORD  IS OMITTED
                      DATA RECORD IS  P-REC.
*
*يوصف غالبا سجل بطول ١٣٢ حرفا وهو عرض الطابعة
*WORKING-STORAGE وتحرك السجلات الأخرى الموصوفة في
*اليه وتتم الكتابة باسم هذا السجل P-REC ومايحتوي
*                            من بيانات مختلفة .
*
 01     P-REC          PIC      X(132).
*
 WORKING-STORAGE SECTION.


*---------------------------------------------------*
*    توصف العناوين كما هو واضح في ورقة    *
*    التوصيف المرفقة PRINT CHART وتسمي    *
* FILLER  العناوين الثابتة او الفراغات    *
*---------------------------------------------------*

  01     HEADER-1.
   03    FILLER    PIC  X(05) VALUE SPACES.
   03    FILLER    PIC  X(15) VALUE
```

```
        '   N A M E      '.
  03    FILLER    PIC  X(4)   VALUE SPACES.
  03    FILLER    PIC  X(15)  VALUE
        ' S A L A R Y  '.
  03    FILLER    PIC  X(11)  VALUE SPACES.
  03    FILLER    PIC  X(16)  VALUE
        'L O C A T I O N '.
  03    FILLER    PIC  XX     VALUE SPACES.
  03    FILLER    PIC  X(08)  VALUE
        '  A G E '.
  03    FILLER    PIC  X(50)  VALUE SPACES.


*-------------------------------------------------------------*
* السطر التالي هو لتوصيف سطر جسم التقرير                        *
*              اي السطر الذي يحتوي على BODY LINE               *
* البيانات المتغيره والذي سيتكرر على طول                        *
*                                                     الصفحه  *
*-------------------------------------------------------------*


 01     DETIL-1.
  03    FILLER    PIC  X(7)   VALUE SPACES.
  03    NAME-P    PIC  A(15).
  03    FILLER    PIC  X(05)  VALUE SPACES.
  03    SAL-P     PIC  9(04).
  03    FILLER    PIC  X(22)  VALUE SPACES.
  03    LOC-P     PIC  X(11).
  03    FILLER    PIC  X(07)  VALUE SPACES.
  03    AGE-P     PIC  99.
  03    FILLER    PIC  X(52)  VALUE SPACES.

*
*       عداد يستخدم لتجميع
*                   عدد الاسطر
 77     L-CONT    PIC   99      VALUE 0.

* الحقول التاليه تستخدم للتسطير بين
*         سطور التقرير على عرض الصفحه
 77     L-H       PIC   X(80) VALUE ALL '='.
 77     L-D       PIC   X(80) VALUE ALL '-'.
```

```
*
*   قســم الإجراءات :
*
 PROCEDURE DIVISION.


*
*   الخطوة الأولى هي فتح الملفات
        OPEN        INPUT    IN-FILE
                     OUTPUT   P-FILE.


*
*   WT-2     طباعة العناوين في الفقرة
*
        PERFORM  WT-2 THRU EXIT-2.

*
 RED-1.
        READ    IN-FILE   AT END
                      GO TO END-JOB.
 ASK-1.
        IF SALARY > 3000 OR SALARY < 2000
                                 GO TO RED-1.
*
 ASK-2.
        IF LOCATION NOT = 'RIYADH      '
                                 GO TO RED-1.
*
 ASK-3.
        IF AGE       NOT < 36      GO TO RED-1.


*
*        IN-FILE تحريك الحقول من ملف الإدخال
*   DETIL-1     المقروء الى حقول سجل الطباعة
*   الموجود في قسم توصيف البيانات الداخلية
 WT-1.
        MOVE        NAME        TO        NAME-P.
        MOVE        SALARY      TO        SAL-P.
        MOVE        LOCATION    TO        LOC-P.
        MOVE        AGE         TO        AGE-P.
```

```
*
*
*   : كتابه السجل على ملف الطباعة
*
        WRITE    P-REC     FROM    DETIL-1
                          AFTER ADVANCING 3 LINES.
        WRITE    P-REC     FROM    L-D.
*
*
*       اضافة 1 لعداد الاسطر  ثم مقارنة
*       محتواه مع العدد 10 لان عدد الاسطر
*       المسموح طباعتها في الصفحة الواحدة
*       هو 10 , فاذا كان اكبر تقلب الصفحة
*       وتطبع العناوين على الصفحة الجديدة
*       ثم يذهب التنفيذ الى قراءة سجل جديد ,
*       واذا لم يكن اكبر فيذهب التنفيذ الى
*       معالجة سجل جديد مباشرة .
*
        ADD      1          TO      L-CONT.
*
        IF       L-CONT     >       10
                 PERFORM   WT-2 THRU EXIT-2.
*
        GO TO  RED-1.
*
*
*   : فقرة قلب الصفحة وكتابة العناوين
*   لاحظ تحريك القيمة صفر لعداد الاسطر
 WT-2.
        MOVE     ZERO       TO      L-CONT.
        MOVE     SPACES     TO      P-REC.
        WRITE    P-REC      AFTER   PAGE.
        WRITE    P-REC FROM L-H    AFTER 1 LINE.
        WRITE    P-REC FROM HEADER-1 AFTER 1.
        WRITE    P-REC FROM L-H    AFTER 1 LINE.
  EXIT-2.
        EXIT.
*
*   : فقرة اغلاق الملفات ونهاية العمل
  END-JOB.
        CLOSE    IN-FILE P-FILE.
        STOP     RUN.
```

شكل مدخلات البرنامج

```
ZUHAIR KHAYAT      2500     RIYADH   27          00000100
ALI ALGHAMDI       2000     RIYADH   29          00000200
RIAD MAHMOOD       5000     RIYADH   20          00000300
SALEH MONAJED      3000     RIYADH   24          00000400
ABDULLAH GHANEM    2500     RIYADH   27          00000500
ALI SALEH          3000     RIYADH   28          00000600
MONEM MAHJOOB      3000     JEDDAH   29          00000700
IBRAHIM ALMOSA     2500     RIYADH   43          00000800
FERAS AHMAD        2000     RIYADH   23          00000900
KHALDOON JNAID     2000     RIYADH   33          00001000
SALEM ABDULLA      2500     RIYADH   32          00001100
IBRAHIM ALHARBI    2500     RIYADH   30          00001200
IBRAHIM AOTAIBI    2400     RIYADH   33          00001300
SABER ALYOUSEF     2400     RIYADH   33          00001400
HAYTHAM ALZOUHD    3000     RIYADH   33          00001500
ISA ALMADANI       3000     RIYADH   33          00001600
KHALED NJAIM       3000     RIYADH   33          00001700
```

٣ - ٦ اسئلة تمرينات

سؤال ١ - بين الخطأ والصحيح من التعليمات التالية المكتوبة بلغة كوبول علماً بأن اسم الملف يتبعه كلمة FILE واسم السجل يتبعه كلمة RECORD :

1. OPEN FILE - ONE, FILE - TWO.
2. OPEN INPUT FILE - ONE, FILE - TWO.
3. OPEN INPUT FILE - ONE, FILE - TWO.
4. OPEN FILE - ONE.
5. OPEN EXTEND FILE - ONE, FILE - TWO.
6. OPEN I - 0 FILE - ONE, FILE - TWO.
7. READ FILE - ONE.
8. READ FILE - ONE, AT END GO TO E - OF - DATA.
9. READ RECORD - ONE AT END MOVE "NO" TO MORE - DATA.
10. READ FILE - ONE AT END CLOSE FILE - ONE, FILE - TWO
    STOP RUN.
11. WRITE RECORD - ONE.
12. WRITE FILE - TWO AFTER 1 LINE.
13. WRITE RECORD - ONE AFTER LINE - NO LINE.
14. WRITE RECORD - TWO AFTER PAGE.
15. CLOSE INPUT FILE - ONE, OUTPUT FILE - TWO.
16. ADD X - FLD 18 Y - FLD GIVING R - FLD
17. ADD X - FLD Y - FLD Z - FLD GIVING R - FLD.
18. ADD A - FLD TO B - FLD.
19. DIVIDE A - FLD INTO 100.
20. DIVIDE 100.25 INTO 4 GIVING R - FLD.

سؤال ٢ ـ لدينا الجزء التالي من برنامج كوبول

```
PROCEDURE DIVISION.
MAIN - PARA.
        PERFORM    A.
        PERFORM    B    3 TIMES.
        PERFORM    C    THRU E.
        PERFORM    F    UNTIL    N = 5.
        STOP RUN.
A.  .
      .
        .
B.  .
      .
        .
C.  .
      .
        .
D.  .
      .
        .
E.  .
      .
        .
F.  .
      .
        .
            ADD 1 TO N.
```

والمطلوب بيان عدد مرات تنفيذ كل فقرة من الفقرات بعد تنفيذ التعليمات الأربعة من PERFORM الواردة في الجزء الرئيسي من البرنامج.

سؤال ٣ -

يوجد في الجدول التالي حقلان : الأول يعتبر المصدر SOURCE والثاني المستلم RECEIVING وقد تم وصف كل من هذين الحقلين و بيان قيمة الحقل المصدر. والمطلوب تحديد القيمة الناتجة حسب الوصف المبين للحقل المستلم؟

SOURCE FIELD		RECEIVING FIELD	
PICTURE	VALUE	PICTURE	EDITED RESULT
9 (6)	345678	9 (4).99	
9 (4)V99	345678	9 (6)	
9 (4)V99	345678	9 (6).99	
9 (4)V99	345678	$ 9 (4) .99	

سؤال ٤ - لدينا البرنامج التالي والمكتوب بلغة كوبول من أجل طباعة جدول بالاشخاص الذين أعمارهم أقل من ٣٠ عاماً وذلك من ملف الموظفين المخزن على بطاقات مثقبة. والمطلوب قراءة هذا البرنامج ودراسته ثم اتمام الجمل أو الكلمات الناقصة واللازمة ليؤدي البرنامج العمل المرسوم له.

```
000100    IDENTIFICATION DIVISION.
000200    PROGRAM - ID. CPROGI.
000210 [          ].    AJND.  ———— (1)
000220 **              THIS PROGRAM WILL PROCESS A FILE OF EMPLOYEE
000230 **              RECORDS AND PRINT THE NAMES UNDER 30 YEARS.
000300    ENVIRONMENT DIVISION.
```

```
000400    CONFIGURATION SECTION.
000500    SOURCE - COMPUTER. IBM - 3033.
000600    OBJECT - COMPUTER. IBM - 3033.
000700    INPUT - OUTPUT SECTION.
000800    FILE - CONTROL.                 (2)
000810           SELECT CARD - FILE ASSIGN TO UT - S - SYSIN.
000900    [    ] PRINT - FILE ASSIGN TO UT - S - PRF.
001000    [      ] ------------------------------------ (3)
001100    FILE SECTION.
001200    FD          PRINT - FILE [        ] IS OMITTED ---- (4)
001300                RECORD CONTAINS   132    CHARACTERS.
001310                DATA RECORD   IS   PRINT - LINE.
001311    01        PRINT - LINE.
001312           05    FILLER        PIC X (1).
001313           05    PRINT - NAME [    ] --------- (5)
001314           05    FILLER        PIC X (2).
001315           05    PRINT - AGE        PIC X (3).
001316           05    FILLER        PIC X (3).
001317           05    PRINT - SALARY    PIC 9 (5).
001318           05    FILLER        PIC X (94).
001320    FD          CARD - FILE LABEL RECORD IS OMITTED
001321                RECORD CONTAINS BO CHARACTERS
001322                DATA RECORD IS EMPLOYEE - CARD.
001330    01        EMPLOYEE - CARD.
001340           05    CARD - NAME        PIC X (25).
001350           05    CARD - TITLE       PIC X (10).
```

```
001360          05   CARD - AGE        PIC 99.
001370          05   FILLER            PIC XX.
001371          05   CARD - SALARY     PIC 9 (5).
001380          05   FILLER            PIC X (36).
001390 *  [        ]----------------------------------(6)
001400    [        ]
001410  77       MORE - INPUT       VALUE 'YES'    PIC XXX.
001740 *
001900  PROCEDURE DIVISION.
002000  MAIN - ROUTINE.
002001 *          HOUSEKEEPING ROUTINE ----------------(7)
002010 [    ] ....INPUT CARD - FILE
0022100              OUTPUT PRINT - FILE.
002110          MOVE SPACES TO PRINT - LINE.
002120          MOVE 'SALARY REPORT FOR EMPLOYEE UNDER 30'
002130           TO PRINT - LINE.
002140          WRITE PRINT - LINE AFTER ADVANCING 2 LINES.
002150          MOVE SPACES TO PRINT - LINE.
002160          WRITE PRINT - LINE AFTER ADVANCING 1 LINE.
002170 *        MAIN - LOOP ROUTINE.
002200          READ CARD - FILE
002210                      AT END MOVE 'NO' TO MORE - INPUT.
002212 [    ] .... PROCESS - EMPLOYEE - RECORDS -------(8)
002220               UNTIL MORE - INPUT  = 'NO'.
002221 *        TERMINATING ROUTINE.
002222          CLOSE CARD - FILE        PRINT - FILE.
```

```
002230          STOP RUN.
002231 *        SUBROUTINES
002290     PROCESS - EMPLOYEE - RECORDS.
002291          IF      CARD - AGE < 30          (9)
002292                  MOVE [    ] TO      PRINT - LINE          (10)
002293                  MOVE CARD - NAME    TO      PRINT - NAME
002294                  MOVE [        ] TO      PRINT - AGE
002295                  MOVE CARD - SALARY    TO      PRINT - SALARY
002296                  WRITE PRINT - LINE AFTER ADVANCING 2 LINES.
002297          READ CARD - FILE
002300                  AT END MOVE [    ] TO MORE - INPUT.
                                         (11)
```

تمرين ١ :

ادخلت المعلومات الخاصة باستهلاك المشتركين في شركة الكهرباء على بطاقات مثقبة بمعدل بطاقة لكل مشترك واسم الملف CARD - F وفرزت هذه البطاقات تصاعدياً على رقم المشترك علماً بأن شكل سجل الادخال كما يلي :

	Field name	Picture
رقم المشترك	C - NO	9 (5)
القراءة الحالية	C - READ	9 (7)
تاريخ القراءة	C - DATE	9 (8)
ملاحظات	C - COMNT	X (3ϕ)
فراغ	C - FLR	X (3ϕ)

والمطلوب : كتابة برنامج يقرأ الملف CARD - F و ينقل محتواه الى ملف موجود على قرص ممغنط اسمه CONSMPFL وشكله كالتالي :

Field name	Picture
CONS - NO	9 (5)
CONS - READ	9 (9)
CONS - DATE	9 (8)
CONS - COMNT	X (3ϕ)
FILLER	X (3ϕ)

تمرين ٢ :

في أحد برامج معهد الادارة تقدم للامتحان ١٠٠ متدرب ، و بعد ظهور النتائج تبين أن عدد الناجحين ٧٠ متدرباً وعدد الراسبين ٣٠ متدرباً .

ادخلت بيانات المتدربين على شريط ممغنط بعد أن فرزت SORTED بحيث كانت سجلات الناجحين أولاً ثم سجلات الراسبين .

والمطلوب : كتابة برنامج بلغة كوبول ينقل الطلاب الناجحين على ملف اخراج اسمه DFILE موجود على قرص ممغنط ، و ينقل الطلاب الراسبين على ملف اخراج آخر اسمه TFILE على قرص ممغنط ، مع ملاحظة ما يلي :

١ ـ طول سجل الإدخال هو ٨٠ حرف

٢ ـ استعمل تعليمة PERFORM لحل هذا البرنامج .

٣ ـ اسم ملف الإدخال هو INFILE .

تمرين ٣ :

لديك ملف اسمه STCKCARD موجود على بطاقات مثقبة وشكل السجل فيه كالتالي :

	Field name	Picture
الكمية	QTY	9 (4) V99
سعر الوحدة	PRICE	9 (3) V99
غير مستخدم	FILLER	X (96)

المطلوب كتابة برنامج يحسب السعر الاجمالي للسلعة وذلك بضرب الكمية بسعر الوحدة، ثم طباعة الكمية وسعر الوحدة والسعر الاجمالي على الطابعة، وفي نهاية العمل اطبع مجموع السعر الاجمالي.



لغة كوبول المتقدمة

الفصل الرابع	التعليمات المتقدمة في لغة كوبول (قسم الاجراءات)
الفصل الخامس	أقسام برنامج كوبول الأخرى والعبارات المتقدمة
الفصل السادس	الجداول في لغة كوبول
الفصل السابع	الفرز والدمج في لغة كوبول
الفصل الثامن	الملفات ومعالجتها
الفصل التاسع	البرمجة الهيكلية بلغة كوبول
الفصل العاشر	تدقيق البرنامج واختياره



الفصل الرابع

التعليمات المتقدمة في لغة كوبول
(قسم الاجراءات)

٤ ـ ١ لمحة عامة :

هـذا الـفـصـل هـو الـفصل الأ ول من القسم الثاني الذي يحتوي على توسعة وتعمق أكثر في تعليمات لغة كوبول التي مرت في القسم الأ ول وكذلك على بعض التعليمات الجديـدة الـتـي تُكمّل التعليمات السابقة لتعطي المبرمج قدرة أكبر على كتابة البرنامج واختيار أفضل الحلول .

فـفي هذا الفصل سوف يتم ذكر تفاصيل أكثر عن تعليمة المقارنة IF والاختبارات المـختلفة المتعلقة بها وكذلك المقارنة المركبة والمتداخلة، ثم ترد تعليمة فحص البيانات INSPECT مـع الإشـارة الى الـتـعـليمة المماثلة EXAMINE، ثم إضافة عبارة التطابق CORRESPONDING على الـتـعـليمات الحسابية وتعليمة النقل. كما سوف يتم شرح تـعـلـيمة عرض البيانات DISPLAY وقبولها ACCEPT واستخدام INTO مع القراءة و FROM مـع الـكـتـابـة وعـبارتي ROUNDED و ON SIZE ERROR وأخيراً يتم بيان تعليمة GO TO المحسوبة (أو المشروطة).

٤ ـ ٢ تعليمة IF

وتـشـتـمـل هـذه التعليمة كما مر معنا المقارنة بين طرفين إضافة إلى شكلها المذكور آنفاً تشمل المقارنات التالية :

٤ ـ ٢ ـ ١ اختبار نوعية بيانات الحقل CLASS TEST

يحتاج المبرمج أحياناً تحديد نوعية البيانات الموجودة في الحقل لذلك فقد زودت لغة كوبول بامكانية المقارنة على هذا الحقل ومعرفة نوع البيانات وتأخذ تعليمة IF عندئذ الشكل العام التالي :

```
IF identifier    IS    [NOT]  { NUMERIC    }
                              { ALPHABETIC }
```

ويجب وصف الحقل المختبر على أنه رقمي بالصورة الرقمية (أي نوع 9 في PIC) ووصف الحقل المراد اختباره على أساس أبجدي ALPHABETIC بالوصف A*. كما يمكن اجراء الاختبار على الحقل الموصوف بالنوع X على كلا الحالتين، والجدول التالي يوضح هذه الصورة بشكل أكثر :

Data Type	Valid Tests
Numeric (9)	NUMERIC, NOT NUMERIC
Alphahumeric (X)	ALPHABETIC, NOT ALPHABETIC
	NUMERIC, NOT NUMERIC, ALPHABETIC, NOT ALPHABETIC

أمثلة : إذا كان لدينا الحقول الثلاثة التالية والموصوفة في ثلاثة أنواع من البيانات :

```
ϕ5   NUMERIC - FLD          PIC 9 (6).
ϕ5   ALPHA - FLD            PIC A (1ϕ).
ϕ5   ALPHA - NUM - FLD      PIC X (5).
```

(*) في معظم المترجمات يشترط استخدام عبارة USAGE مع DISPLAY في توصيف الحقل المراد اختباره بعد عبارة PICTURE.

فإنه يمكن كتابة تعليمة الاختبار IF على النحو التالي :

```
IF   NUMERIC - FLD     IS   NUMERIC ....
IF   NUMERIC - FLD     IS   NOT NUMERIC ........
IF   ALPHA - NUM - FLD   IS   NOT ALPHABETIC ...........
IF   ALPHA - NUM - FLD   IS   NUMERIC .......
```

ولكن لا يمكن كتابة التعليمات التالية لعدم صلاحيتها :

```
IF   ALPHA - FLD     IS   NOT NUMERIC
IF   NUMERIC - FLD   IS   NOT ALPHABETIC
```

ويجب التنويه أخيراً الى أن وجود الإشارة الجبرية في الحقل الرقمي لا يسمح باختباره على أنه رقمي.

٤ ـ ٢ ـ ٢ اختبار الاشارة الجبرية SIGN TEST

إن هذا النوع من الاختبار بتعليمة IF يحدد اشارة الحقل الرقمي، والشكل العام يأخذ الصورة التالية :

```
     ⎧ identifier ⎫  ⎧ IS   [NOT] POSITIVE ⎫
IF   ⎨ arithmetic ⎬  ⎨ IS   [NOT] NEGATIVE ⎬
     ⎩ expression ⎭  ⎩ IS   [NOT] ZERO     ⎭
```

وتكون قيمة الحقل identifier موجبة Positive إذا كان أكبر من الصفر وتكون سالبة إذا كانت أقل من الصفر. و يستخدم هذا الاختبار للتحقق من قيمة البيانات المدخلة أو قيمة البيانات المحسوبة الناتجة عن عملية حسابية.

أمثلة :

1) IF NET - PAY IS NOT POSITIVE PERFORM CAL - TAXES.

2) IF BALANCE IS NEGATIVE PERFORM OVER - DUE.

٤ ـ ٢ ـ ٣ تعريف واختبار اسم الشرط CONDITION NAME TESTS

إن عبارة الشرط CONDITION في تعليمة IF غالباً ما تختبر قيمة الرمز الداخل الى الحاسب (أو إلى البرنامج) مثل :

IF YR - CODE = " 1" ...

إذا كان مثل هذا الرمز مسموح به، إلا أن معنى هذا الرمز في الحقل YR - CODE لا يكون واضحاً وغير معروفاً في البرنامج. لهذا تضمنت لغة كوبول بديلاً لهذا الشرط يدعى اسم الشرط ليعطي هذه اللغة قوة أكبر في التوثيق والايضاح. وهذا الاسم يظهر في المستوى Level رقم 88 و يعيد توصيف الحقل وتحديد القيمة الممكن أن ترد فيه دون أن يعطي للحقل أي حجم أو وصف آخر. كما يجب استعمال هذا التعريف فقط مع الحقول الأولية elementry أي التي تحتوي على عبارة (PIC).

مثال :

```
ϕ5   YR - CODE PIC X.
     88   GEORGIAN        VALUE " 1 ".
     88   HAGRI        VALUE " 2 ".
     88   OTHER        VALUE " 3 ".
     88   VALID - YR - CODE VALUE "1",   "2",   "3".
```

فإذا وجدت هذه التوصيفات في قسم وصف البيانات DATA DIVISION فإن المقارنة تصبح كما يلي :

IF GEORGIAN PERFORM M - S.

وهي تعادل :

```
IF YR - CODE = "1" PERFORM  M - S.
```

وهناك ثلاثة فوائد من اسم الشرط وهي :

- زيادة في الايضاح وتوثيق البرنامج .
- تسهيل تعديل وتطوير البرنامج وامكانية إضافة رموز أخرى على الحقل دون إجراء تعديل كبير على البرنامج .
- السماح بتجميع عدة رموز بتعريف واحد في المستوى 88 .

وبشكل أكثر تفصيلاً يمكن وضع الصيغة العامة لتعريف اسم الشرط على النحو التالي :

```
                 { VALUE IS   }
88 data - name   {            }   literal - 1 [THRU literal - 2]
                 { VALUES ARE }

                              [literal - 3 [THRU literal - 4 ]....
```

مثال :

```
ϕ3  NATION - CODE        PIC 9.
    88  SAUDI          VALUE 1.
    88  ENGLISH        VALUE 2.
    88  TURKISH        VALUE 3.
    88  PAKISTAN       VALUE 4.
    88  AFRICAN        VALUES ARE 5 THRU 1ϕ.
    88  AMERICAN       VALUES ARE 20 THRU 26.
    88  AUSTRALIAN     VALUES ARE 3ϕ , 33.
```

وتعليمة : IF ENGLISH GO TO E - RTN.

تعادل : IF NATION - CODE = 2 GO TO E - RTN.

٤ - ٢ - ٤ المقارنة المركبة COMPOUND IF

يمكن جمع أكثر من شرط (مقارنة) بتعليمة IF حيث تشكل فيما بينها مقارنة مركبة و يستخدم لهذا الغرض المعاملات المنطقية* AND وOR وNOT . وعندئذ يمكن أن تأخذ تعليمة IF المركبة الشكل العام التالي :

1)

```
IF Condition - 1  [ { AND }  Condition - 2 ] ....
                    { OR  }

   { Statement - 1    }  [ ELSE { Statement - 2    } ]
   { NEXT SENTENCE    }  [      { NEXT SENTENCE    } ]
```

2)

```
IF NOT Condition { Statement - 1  }  [ ELSE { Statement - 2  } ]
                 { NEXT SENTENCE  }  [      { NEXT SENTENCE  } ]
```

حيث يقصد بكلمة : Condition الشرط ويمكن أن يكون على النحو التالي** :

- مقارنة بسيطة Simple Condition
- شرط بسيط منفي Negated Simple Condition
- شرط مركب Compound Condition
- شرط مركب منفي Negated Compound Condition
- مجموعة من الشروط البسيطة أو المركبة

(*) Logical Operators

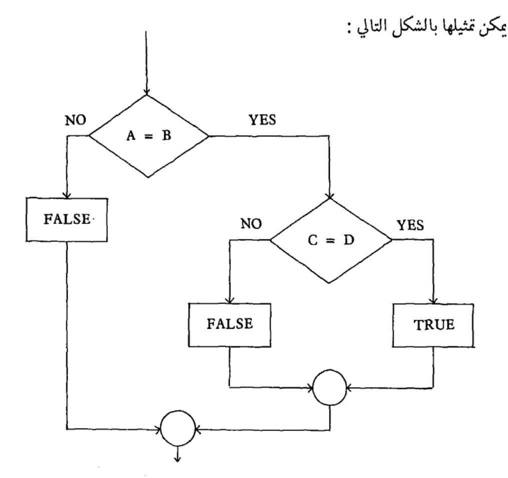
(**) TI990 COBOL

فالعامل المنطقي AND يعني أنه إذا تحقق الشرط الأول والثاني البسيطين أو المركبين في تعليمة IF فان النتيجة للمقارنة الكلية تكون (True) محققة.

فالمقارنة المركبة التالية :

IF A = B AND C = D .....

يمكن تمثيلها بالشكل التالي :



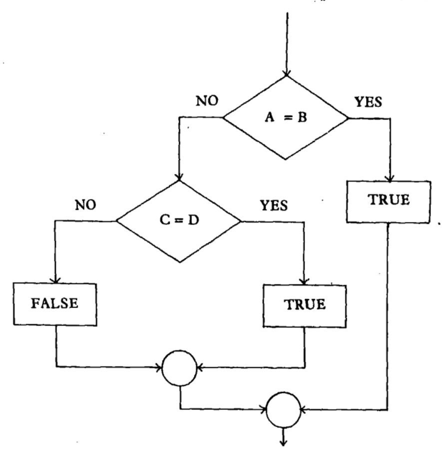
وإذا لم يتحقق أحد الشرطين (A = B) أو (C = D) فان نتيجة IF تكون (False) غير محققة.

أما العامل المنطقي OR فيعني أن تحقق أحد الشرطين المذكورين في تعليمة IF يكفي لتكون نتيجة المقارنة True.

فالمقارنة المركبة :

IF A = B OR C = D ....

يمكن تمثيلها بالشكال التالي :



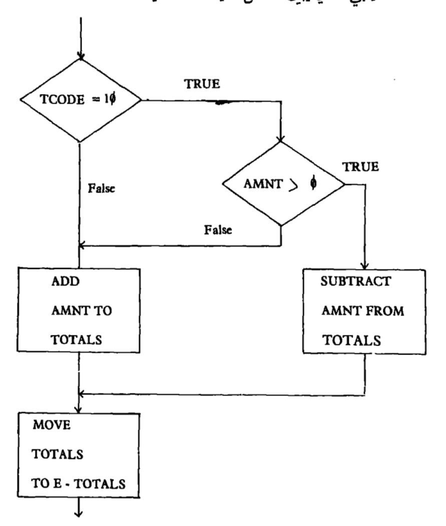
وبموجب استخدام العامل OR فان تحقق أحد الشرطين (A = B) أو (C = D) يؤدي الى الحصول على النتيجة الصحيحة True في الرسم وعدم تحقق كليهما معاً ينفي تحقق النتيجة المرجوة من المقارنة .

أما العامل المنطقي NOT فهو اذا أضيف الى تعليمة IF فيكون عدم تحقق الشرط هو النتيجة المطلوبة من المقارنة True وتحقق الشرط يؤدي الى نقل التسلسل إلى جهة False .

فمثلاً تعليمة IF التالية توضح ذلك :

```
IF NOT  (TCODE IS  =  10 AND AMNT IS > ϕ)
            ADD AMNT TO TOTALS
    ELSE SUBTRACT AMNT FROM  TOTALS .
MOVE TOTALS TO E - TOTALS.
```

والمخطط الانسيابي التالي يبين تسلسل تنفيذ هذه التعليمات :



و يلاحظ عند استخدام المعاملات المنطقية امكانية استخدام الأقواس من أجل تحديد المقارنات والشروط في تعليمة IF لأن أولوية تنفيذها تلعب دوراً كبيراً عند عدم وجود الأقواس حيث أن ترتيب تنفيذ العمليات المختلفة في تعليمة IF الواحدة تكون وفق التسلسل التالي :

١ - الشروط البسيطة (أو المعاملات العلائقية) Relational Operators.
٢ - العامل المنطقي NOT
٣ - العامل المنطقي AND (و يبدأ من جهة اليسار إلى اليمين في حال وجود أكثر من معامل).
٤ - العامل المنطقي OR (و يبدأ من جهة اليسار إلى اليمين في حال وجود أكثر من معامل).

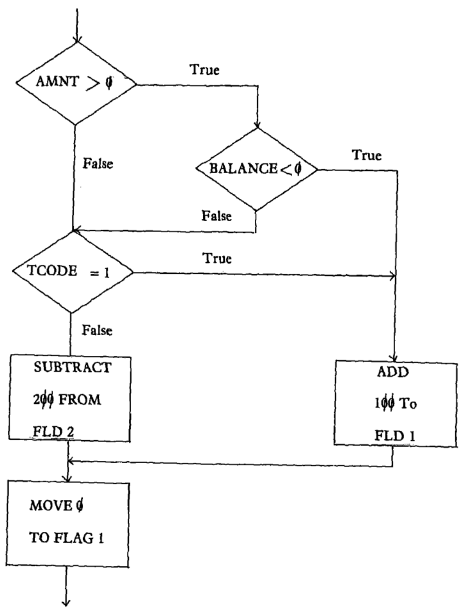
أمثلة :

التعليمات التالية تبين استخدام AND مع OR :

```
IF     TCODE = 1     OR AMNT > ϕ AND BALANCE < ϕ
                     ADD     1ϕϕ     TO FLD 1

     ELSE
                SUBTRACT 2ϕϕ FROM FLD 2.
MOVE     ϕ     TO     FLAG 1.
```

وفيما يلي المخطط الانسيابي لهذه التعليمات :



والتعليمات التالية تبين استخدام AND مع OR بوجود أقواس لحصر الشرط المستخدم مع OR :

IF TCODE = 1 AND (AMNT > ∅ OR BALANCE < ∅)

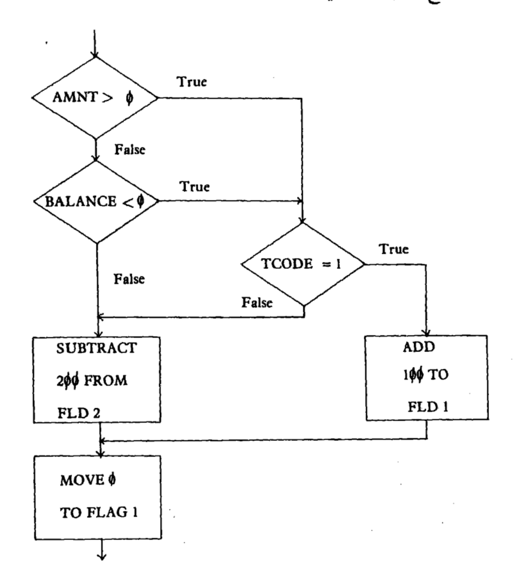
ADD 1∅∅ TO FLD 1

ELSE

SUBTRACT 2ϕϕ FROM FLD 2.

MOVE ϕ TO FLAG 1.

و يبين أيضاً المخطط الانسيابي التالي شكل التسلسل المنطقي لتنفيذ هذه التعليمات مع ملاحظة الفرق بين هذا المثال والمثال السابق بسبب وجود الأقواس.



٤ ـ ٢ ـ ٥ تعليمة IF المتداخلة NESTED IF

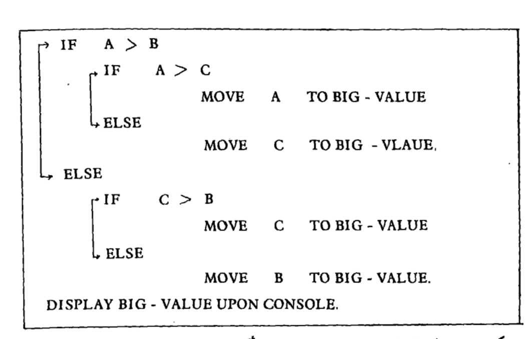
إن الشكل العام لتعليمة IF هو ما يلي :

```
                ⎧ Statement - 1      ⎫ ⎡      ⎧ Statement - 2      ⎫ ⎤
IF Condition    ⎨                    ⎬ ⎢ ELSE ⎨                    ⎬ ⎥
                ⎩ NEXT SENTENCE      ⎭ ⎣      ⎩ NEXT SENTENCE      ⎭ ⎦
```

حيث أن الشرط Condition يمكن أن يكون أي نوع من المقارنة (الاختبارات Tests) مثل مقارنة النوع Class أو اسم الشرط Condition - name أو مقارنة بسيطة أو مقارنة الاشارة الجبرية أو مقارنة مركبة. وعبارة NEXT SENTENCE يقصد بها متابعة تنفيذ البرنامج بعد النقطة الموجودة في آخر تعليمة IF .

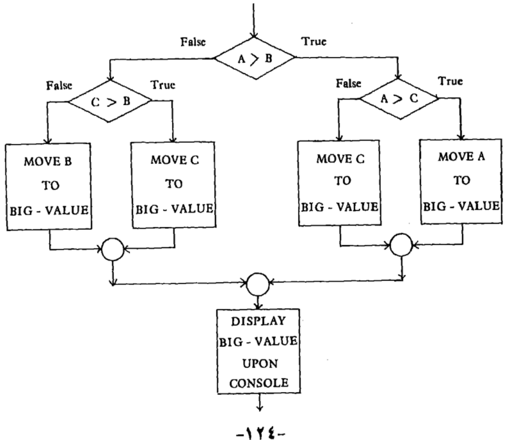
أما تعليمة IF المتداخلة (المتشابكة) فهي عبارة عن كون أي من العبارات Statement - 1 أو Statement - 2 بحد ذاتها تعليمة IF أخرى. أو بمعنى آخر هناك أكثر من تعليمة IF في جملة واحدة.

ويجب على المبرمج ملاحظة أن كل تعليمة IF يتبعها عبارة ELSE الخاصة بها وتكون كل كلمة ELSE في الجملة تابعة لأقرب IF لها حسب قاعدة الترجمة بلغة كوبول Compilation. وعندما لا يوجد لكل IF عبارة ELSE فيجب على المبرمج تحديد النهاية المنطقية لـ IF لمعرفة تبعيتها لأي عبارة ELSE . ولأجل سهولة الكتابة والمراجعة تكتب تعليمة IF المتداخلة على شكل مدرج من جهة اليسار لتحديد تسلسلها المنطقي وتحديد عبارة ELSE التابعة لها . والشكل التالي يعطي مثالاً على ذلك :



```
IF    A > B
      IF     A > C
                   MOVE    A    TO BIG - VALUE
      ELSE
                   MOVE    C    TO BIG  - VLAUE.
ELSE
      IF     C > B
                   MOVE    C    TO BIG - VALUE
      ELSE
                   MOVE    B    TO BIG - VALUE.
DISPLAY BIG - VALUE UPON CONSOLE.
```

ويمكن رسم هذه التعليمات التي تقوم باختيار أكبر قيمة من بين ثلاث قيم على التوالي :

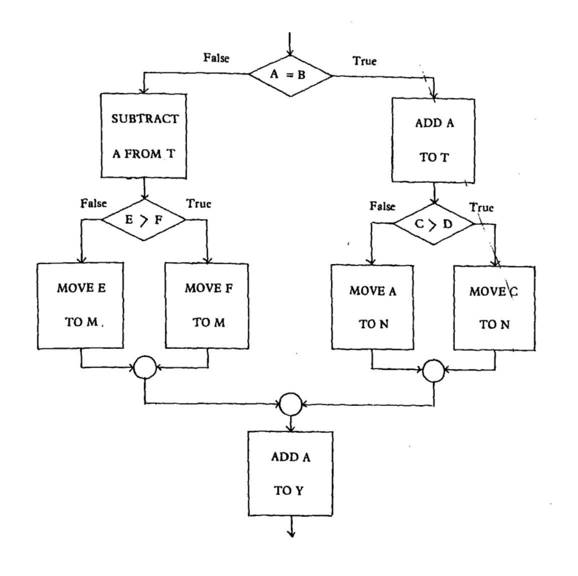


أما التعليمات التالية فهي تمثل مثال آخر على استخدام IF المتداخلة :

```
IF    A = B ADD    A    TO    T
      IF    C > D MOVE    C    TO N
      ELSE
            MOVE    A    TO    N
ELSE
      SUBTRACT    A    FROM    T
      IF    E > F    MOVE    F    TO    M
      ELSE
            MOVE    E    TO    M.
ADD    A    TO    Y.
```

و يلاحظ من المثالين أن جملة IF المتداخلة تنتهي بنقطة واحدة Period في نهاية آخر مقارنة ويجب على المبرمج الانتباه الشديد لوجود أية نقطة أخرى ضمن المقارنات المتداخلة في الجملة الواحدة لأنها سوف تسبب خللاً في منطق تسلسل المقارنات . و يستعان عادة بالمخطط الانسيابي لتوضيح سير المقارنات و يشار الى الجهة التي تحقق الشرط الموضوع في المقارنة بكلمة True أو Yes والى الجهة الأخرى (عدم تحقق الشرط) بكلمة False أو No .

والرسم التالي يبين سير المخطط الانسيابي للتعليمات السابقة الذكر.



٤ ـ ٣ استخدام عبارة CORRESPONDING

في الحالات العادية يوّصف المبرمج أسماء المتغيرات والحقول والسجلات في برنامجه بشكل لا يسمح بتكرار الاسم الواحد لأكثر من مرة أي أن أسماء المتغيرات تكون فريدة UNIQUE ، ولكن مع تطور لغة كوبول وظهور النسخ الجديدة منها أدى الى السماح بتكرار أسماء الحقول فقط ضمن السجلات التي تحمل أسماء مختلفة وذلك من أجل زيادة فعالية هذه اللغة في نقل الحقول من سجل لأخر ولتوفير الجهد في كتابة

تـعليمة MOVE بشكل خاص. فمثلاً لو كان لدينا السجلان التاليان في قسم البيانات :

```
ϕ1   CARD - REC.
     ϕ5   STD - NAME                       PIC   (25).
     ϕ5   STD - ADDRESS.
          1ϕ   STREET                      PIC   X (15).
          1ϕ   CITY - NAME                 PIC   A (15).
     ϕ5   STD - SCORES                     PIC   999.
     ϕ5   STD - GRADE                      PIC   X.
     ϕ5   FILLER                           PIC   X (21).
ϕ1   .
       .
         .
           .
             .
ϕ1 PRINT - LN.
     ϕ3   STD - NAME                       PIC   A (25).
     ϕ3   FILLER                           PIC   X (4).
     ϕ3   STD - ADDRESS.
            07 STREET                      PIC   X (15).
            07 CITY - NAME                 PIC   A (15).
     ϕ3   FILLER                           PIC   X (3).
     ϕ3   STD - GRADE                      PIC   X .
     ϕ3   FILLER                           PIC   X (3).
     ϕ3   STD - SCORES                     PIC   999.
```

```
ϕ3   FILLER                      PIC   X (5).
ϕ3   SPECIAL - NUMBER            PIC   9999.
ϕ3   FILLER                      PIC   X (54).
```

فإن لغة كوبول تسمح بوجود نفس الاسم في السجلين للحقل الواحد، ولكن يجب الإشارة الى تبعية هذا الحقل عند استخدامه في قسم الاجراءات فمثلاً لو أردنا جمع الحقل STD - SCORES فيجب الاشارة الى تبعيته في تعليمة الجمع ADD :

```
ADD STD - SCORES IN CARD - REC TO TOTAL - SCORES.
```

وتستخدم للدلالة على التبعية Qualification كلمة IN أو كلمة OF مثل :

```
MOVE STD - GRADE OF CARD - REC TO STD - GRADE OF
    PRINT - LN.
```

ويمكن أن تكون التبعية على أكثر من مستوى مثل :

```
MOVE STREET OF STD - ADDRESS OF CARD - REC TO STREET IN
STD - ADDRESS IN PRINT - LN.
```

كما أنه يمكن الاشارة الى تبعية الأصلية في السجل فقط فيكون المثال السابق :

```
MOVE STREET IN CARD - REC TO STREET IN PRINT - LN
```

وهذا يعادل نفس النتيجة المذكورة أعلاه.

واختيار نفس الأسماء في أكثر من سجل يوفر على المبرمج الوقت اللازم لايجاد أسماء فريدة لكل حقل من الحقول، كما يساعد في نقل أو تداول هذه الحقول بشكل مختصر أكثر عند استخدام كلمة CORRESPONDING أو CORR في التعليمة. وتكون

تعليمة النقل MOVE عندئذ على النحو التالي :

```
         ( CORRESPONDING )
MOVE     {               }    identifier - 1 TO identifier - 2.
         ( CORR          )
```

و يلاحظ أن كلمة CORR هي اختصار لكلمة CORRESPONDING . و باستخدام هذا الشكل لتعليمة النقل MOVE نستطيع نقل (أو نسخ) الحقول في السجل CARD - REC الى الحقل في السجل PRINT - LN بتعليمة واحدة. أي :

MOVE CORR CARD - REC TO PRINT - LN.

حيث تقوم بنقل جميع محتويات الحقول التي تحمل نفس الأسم في السجل CARD - REC الى الحقول نفسها في السجل PRINT - LN. وهذه التعليمة تعادل التعليمات التالية :

```
MOVE   STD - NAME IN CARD - REC TO STD - NAME IN PRINT - LN.
MOVE   STREET IN CARD - REC TO STREET IN PRINT - LN.
MOVE   CITY - NAME IN CARD REC TO CITY - NAME IN
    PRINT - LN.
MOVE     STD - SCORES IN CARD - REC TO STD - SCORES IN
    PRINT - LN.
MOVE     STD - GRADE IN CARD - REC TO STD - GRADE IN
    PRINT - LN.
```

وتجب الاشارة الى أن الحقول التي ليست مشتركة في الاسم بين السجلين تستثنى من عملية النقل. مثل الحقل SPECIAL - NUMBER .

و يستخلص من عبارة التطابق الاختيارية النقاط التالية عند استخدامها :

١ - يتم النقل بين السجلين بوساطة MOVE للحقول الأولية elementary ويجب أن يكون هناك في السجلين حقل واحد متطابق في كل منهما .

٢ - يتم النقل للحقول المتطابقة في الاسم بين السجلين بغض النظر عن ترتيب الحقل في السجل و يعتبر اسم الحقل معيار التطابق.

٣ - الحقول المتطابقة في الاسم يجب أن لا تحتوي العبارات التالية في توصيفها :

REDEFINES , RENAMES , OCCURS , USAGE IS INDEX.

لأنها عندئذ لا يتم نقلها بعبارة التطابق مع MOVE* .

وإضافة الى امكانية استعمال كلمة CORR مع تعليمة MOVE فانه يمكن استعمالها بنفس الاسلوب مع تعليمة الجمع ADD وكذلك الطرح SUBTRACT** .

و يكون الشكل العام لهذين التعليمتين عندئذ كما يلي :

```
ADD  { CORRESPONDING }  identifier - 1 TO identifier - 2
     { CORR          }
     [ROUNDED]   [ON SIZE ERROR imp - statement]

SUBTRACT  { CORRESPONDING }  identifier - 1 FROM identifier - 2
          { CORR          }
     [ROUNDED]   [ON SIZE ERROR imp - statement]
```

(*) GRAUER AND CRAWFORD, PP 118.

(**) ينصح بالرجوع لمترجم كوبول المتوفر للحاسب المستخدم لمعرفة شمول عبارة التطابق لهذه التعليمات .

٤ ـ ٤ تعليمة القراءة READ INTO

يمكن كتابة تعليمة القراءة READ التي سبق ذكرها في الشكل التالي العام :

READ file - name RECORD INTO identifier AT END imperative statement.

واضافة كلمة INTO على تعليمة القراءة تمكن المبرمج من قراءة السجل من الملف الموصوف في هذه التعليمة ونقل محتويات السجل الى المكان identifier الموصوف في قسم التوصيف الداخلي للبرنامج أي WORKING - STORAGE SECTION . والمثال يوضح استخدام هذه التعليمة :

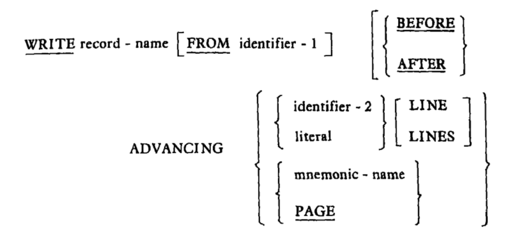
```
FD    CARD - FILE  ...
      .
        .
          .
ϕ1    CARD - REC          PIC X (8ϕ).
      .
        .
          .
WORKING - STORAGE SECTION.
ϕ1    WORK - REC - AREA          PIC X (8ϕ).
      .
        .
          .
PRODCEDURE DIVISION.

          READ CARD - FILE INTO WORK - REC - AREA AT END
               PERFORM END - OF - READ.
```

وبعد تنفيذ تعليمة القراءة تكون محتويات السجل في الحقل WORK - REC - AREA WORK - REC - AREA أي أن هذه التعليمة تشمل تعليمة النقل MOVE.

٤ ـ ٥ تعليمة الكتابة WRITE FROM

قياساً على تعليمة القراءة الى منطقة معينة يوجد في لغة الكوبول تعليمة الكتابة الى الملف من منطقة معينة في الذاكرة وهي WRITE FROM والتي تشتمل أيضاً على تعليمة MOVE ضمناً. فبعد أن يصبح السجل جاهزاً للكتابة على ملف الاخراج في منطقة العمل الداخلي في البرنامج يمكن كتابته ونقله في نفس التعليمة. وتأخذ التعليمة الشكل العام :



وتفيد هذه التعليمة بشكل خاص عند كتابة عناوين أعلى الصفحة HEADERS على ملف الطباعة. والمثال يوضح ذلك :

```
FD    PRINT - FILE  ...
   .
    .
     .
      .
ϕ1    P-REC          PIC X (132).
   .
    .
     .
      .
WORKING - STORAGE SECTION.
```

```
ϕ1     HEADER - LINE - 1.
   ϕ5     FILLER          PIC X (30) VALUE SPACE.
        .
          .
            .
PROCEDURE DIVISION.
      .
        .
          .
   WRITE    P-REC FROM HEADER - LINE - 1 AFTER ADVANCING
   T - OF - PG LINES.
```

وهذه التعليمة WRITE FROM تعادل التعليمتين التاليتين :

```
MOVE    HEADER - LINE - 1 TO P - REC.
WRITE   P - REC AFTER T - OF - PG LINES.
```

٤ ـ ٦ التقريب وكشف الخطأ في التعليمات الحسابية

ROUNDED and ON SIZE ERROR CLAUSES

إن التقريب في الأعداد الحسابية يعتبر ضرورياً خاصة في الأعمال التجارية لتتوافق مع الاجزاء النقدية الصغيرة ، لذلك احتوت لغة الكوبول على امكانية التقريب باضافة كلمة ROUNDED على التعليمات الحسابية المعروفة في تعليمة الجمع أو الطرح أو الضرب أو التقسيم أو تعليمة COMPUTE . و يكون موضع هذه الكلمة في التعليمة الحسابية الجمع أو الطرح أو الضرب أو القسمة كما يلي :

```
{ arithmetic }  [ ROUNDED ]
{ Statement  }
```

وعند وجود كلمة الباقي في التقسيم REMAINDER فانها تأتي بعد كلمة ROUNDED . أي :

DIVIDE .... [ROUNDED]

[REMAINDER] identifier.

أما في تعليمة COMPUTE فتأتي على النحو التالي :

COMPUTE identifier - 1 [ROUNDED] = arithmetic expression

وبكلمة ROUNDED تكون النتيجة الحسابية التي تحتوي على جزء عشري أكثر دقة لأنها تؤدي الى التقريب في النتيجة الى أقرب موضع في الجزء العشري حسب قواعد التقريب في التعليمات الحسابية العادية . أي أن الرقم العشري الأكثر من ٤ يقرب الى القيمة ١ والرقم العشري المساوي أو أقل من ٤ يقرب الى القيمة صفر.

وليكن لدينا الحقلان FLD - 1 و FLD - 2 موصوفين على النحو :

```
ϕ5   FLD - 1   PIC   99V999   VALUE   2.857
ϕ5   FLD - 2   PIC   99V999   VALUE   15.142
```

والحقل FLD - 3 موصوف على النحو :

```
ϕ5   FLD - 3   PIC   99V99
```

ثم نفذت تعليمة الجمع ADD المكتوبة وفق الشكل :

```
ADD   FLD - 1, FLD - 2 GIVING FLD - 3 ROUNDED.
```

فتكون نتيجة الجمع 18,ϕϕ بدلا من القيمة 17.99

أما من أجل كشف الخطأ في العمليات الحسابية فانه يمكن استخدام الجملة ON SIZE ERROR مع التعليمات الحسابية المعروفة في لغة كوبول وتمكن هذه الجملة المبرمج التأكد من أن الحقل الذي سوف يخزن فيه (يحفظ) النتيجة ذو حجم كافي لها أو

أنها تزيد عن عدد البايت المحجوزة (المعرفة) في قسم البيانات. وهذه الجملة في التعليمة الحسابية تقوم بمقام التعليمة IF في حال حصول عدم تطابق في حجم النتيجة المحسوبة وحجم الحقل فان تسلسل البرنامج ينتقل الى التعليمة الأمرية Imperative Statement الموصوفة بعد هذه الجملة من أجل اعطاء المبرمج فرصة التأكد من صحة حساباته. و يكون موضع هذه الجملة في التعليمات الحسابية الجمع أو الطرح أو الضرب أو التقسيم على الشكل التالي :

{ arithmetic statement } [ON SIZE ERROR] { imperative statement }

وفي حال اجتماع جملة التقريب مع جملة كشف الخطأ في حجم الحقل تأخذ التعليمات الحسابية الشكل التالي :

{ arithmetic statement } ROUNDED [ON SIZE ERROR] { imperative statement }

أما تعليمة التقسيم فيمكن أن تكون على النحو التالي :

DIVIDE ... [ROUNDED] [REMAINDER identifier] [ON SIZE ERROR] { imperative statement }

بينما تعليمة احسب COMPUTE تصبح حسب الشكل العام التالي :

COMPUTE identifier - 1 [ROUNDED] = arithmetic expression ...

[ON SIZE ERROR] { imperative statement }

فليكن لدينا الحقول التالية :

```
ϕ5   HOURLY - RATE      PIC 99 VALUE 25.
ϕ5   HOURS              PIC 99 VALUE 4ϕ.
ϕ5   GROSS - PAY        PIC 999.
```

وتعليمة احسب التالية :

```
COMPUTE GROSS - PAY  = HOURLY - RATE * HOURS
                               ON SIZE ERROR
                         PERFORM ERROR - RTN.
```

فان عملية ضرب الحقلين تعطي النتيجة 1ϕϕϕ ولكن الحقل GROSS - PAY معرف على أنه ثلاثة أرقام فقط (PIC 999) ويخزن فيه القيمة ϕϕϕ وهذه النتيجة غير صحيحة حسابياً لذلك فان جملة ON SIZE ERROR توجه سير تسلسل البرنامج الى الروتين المذكور بعدها والمعرف من قبل المبرمج ليأخذ الاحتياطات اللازمة في مثل هذه الحالة .

٤ ـ ٧ تعليمة نقل التسلسل المحسوبة GO TO DEPENDING ON

عرفنا في السابق تعليمة نقل التسلسل البسيطة GO TO التي تقوم بنقل تسلسل تنفيذ البرنامج الى الفقرة المذكورة بعدها . ولكن اذا كان لدينا عمليات نقل تسلسل نعتمد على قيمة معينة مثل :

```
IF   CODE - S  = 1   GO   TO   RTN - A.
IF   CODE - S  = 2   GO   TO   RTN - B.
IF   CODE - S  = 3   GO   TO   RTN - C.
     GO TO ERROR - RTN.
```

حيث أن عملية نقل التسلسل الى الفقرات RTN - A و RTN - B و RTN - C تعتمد على قيمة الحقل CODE - S ولتسهيل عملية البرمجة في كوبول فقد احتوت اللغة على تعليمة GO TO مع عبارة (جملة) DEPENDING ON ويمكن كتابة المثال السابق كما يلي :

GO TO RTN - A , RTN - B , RTN - C DEPENDING ON CODE - S.

GO TO ERROR - RTN.

حيث أن القيمة «1» في الحقل CODE - S تشير الى اسم الفقرة الأولى RTN - A و «2» تشير الى اسم الفقرة RTN - B وهكذا ... وإذا كانت قيمة الحقل CODE - S مساوية للصفر أو سالبة أو أكبر من عدد الفقرة المعرفة بعد GO TO ينتقل تسلسل التنفيذ الى الجملة التالية أي :

GO TO ERROR - RTN.

وتأخذ تعليمة نقل التسلسل المحسوبة الشكل العام التالي :

GO TO paragraph - 1, paragraph - 2 paragraph - 3 ------ paragraph - name - n

DEPENDING ON data - name.

حيث أن paragraph - i تشير الى اسم الفقرة المراد نقل التسلسل اليها و data - name اسم الحقل الرقمي الذي يجب أن يحتوي على قيمة صحيحة integer تدل على اسماء الفقرات المذكورة بعد GO TO .

والحد الأعلى لعدد الفقرات يختلف باختلاف نوع الحاسب المستخدم إلا أنه يتجاوز ١٠٠٠ فقرة(*) .

(*) Stern Stern, STRUCTURED COBOL PROGRAMMING P. 320.

بعض الحاسبات يصل الى ٢٠٣١ فقرة .

٤ ـ ٨ تعليمة عرض البيانات DISPLAY

إن تعليمة عرض البيانات DISPLAY تتيح للمبرمج إظهار بيانات بدون تحديد توصيف لها في قسم البيانات ولا تحتاج الى توصيف ملف اخراج OUTPUT في الجزء الخاص بتعريف الملفات والشكل العام لهذه التعليمة حسب قواعد ANSI هو :

```
           { identifier - 1 }   [ { identifier - 2 } ]
DISPLAY    {                }   [ {                } ] - - - - - - - - - -
           { literal - 1    }   [ { literal - 2    } ]

                   [ UPON mnemonic - name ]
```

أمثلة :

1) DISPLAY STUDENT - NAME.

2) DISPLAY "NAME IS = " STUDENT - NAME.

3) DISPLAY STUDENT - NAME, STUDENT - NO.

4) DISPLAY FIELD - A UPON CONSOLE

في المثال (١) يتم اظهار محتوى الحقل المسمى STUDENT - NAME على الطابعة لعدم ذكر وتحديد الوحدة المراد استخدامها في أظهار البيانات. أما المثال (٢) يقوم الحاسب بإظهار عبارة " = NAME IS" قبل إظهار محتوى الحقل المسمى STUDENT - NAME. والمثال (٣) يتيح للحاسب إظهار محتوى حقلين، وكل هذه البيانات تظهر على الطابعة أيضاً لعدم ذكر رمز الوحدة المطلوبة.

وفي المثال رقم (٤) وباستخدام عبارة UPON الاختيارية يقوم الحاسب باظهار وعرض البيانات على الوحدة المحددة بعد كلمة UPON ، وهي وحدة CONSOLE (ـ وحدة التوجيه والتحكم في الحاسب).

ويمكن توصيف وحدات إظهار أخرى مثل I / 0 TYPEWRITER أو آلة طابعة فرعية ويجب الرجوع في تحديد رمز هذه الوحدة إلى مراجع الشركة الصانعة للحاسب، لأنها تختلف باختلاف نوع الحاسب المستخدم.

إضافة الى الشكل العام المعتمد من قبل هيئة ANSI ولضرورات التعامل مع الشاشة الضوئية الحديثة بوساطة برنامج كوبول يوجد شكل آخر لتعليمة عرض البيانات وهذا الشكل(*) يعتمد على غالبية الحاسبات المسماة MINI - COMPUTER والتي ترتبط بشاشات مرئية مباشرة من أجل التشغيل المباشر ON - LINE.

```
DISPLAY  { identifier - 1 }  [ , UNIT { identifier - 2 } ]
         { literal - 1    }  [        { literal - 2    } ]

   [ , LINE , { identifier - 3 } ]  [ , POSITION { identifier - 4 } ]
   [          { literal - 3    } ]  [            { literal - 4    } ]

   [ , SIZE { identifier - 5 } ]  [ , BEEP ] [ , ERASE ]  [ [ , LOW  ] ]
   [        { literal - 5    } ]                          [ [ , HIGH ] ]

   [ , { identifier - 6 } ]  , UNIT { identifier - 7 } ]  [ , LINE { identifier - 8 } ]
   [   { literal - 6    } ]         { literal - 7    } ]  [        { literal - 8    } ]

   [ , POSITION { identifier - 9 } ]  [ , SIZE { identifier - 10 } ]
   [            { literal - 9    } ]  [        { literal - 10    } ]
```

(*) TI 990

[, BEEP] [,ERASE,] [{ , LOW / , HIGH }] ....

إن الثوابت (literal - 1 literal - 6 ....) يمكن أن تكون أي ثابت شكلي constant figurative . (أي ... SPACE, ZERO)

إن تعليمة العرض هذه تمكن المبرمج من عرض أي متغير أو ثابت(identifier - 1 أو literal - 1 أو identifier - 6 literal - 6 ، ...) على الشاشة الضوئية .

عندما تحتوي تعليمة DISPLAY على أكثر من متغير ثابت لأجل عرضها أو إظهارها فإن ذلك يتم على التتالي حسب ترتيبهم بعد كلمة DISPLAY .

عبارة UNIT اذا ذكرت فإنها يجب أن تكتب قبل بقية العبارات الأخرى الاختيارية ، والقيمة التي بعد كلمة UNIT تُوصف (أو تحدد) رقم الوحدة المستخدمة في إظهار البيانات (أي الشاشة الضوئية التي تعرّف أحيانا برقم خاص بها) . وإذا لم تذكر عبارة UNIT فان عملية الاظهار في التعليمة تتم على نفس الشاشة التي ينفذ عليها البرنامج .

عبارة LINE تحدد رقم السطر المراد استخدامه في عرض البيانات على الشاشة (تحتوي الشاشة عادة على ٢٤ سطراً) . وإذا كانت قيمة رقم السطر أكبر من عدد الأسطر على الشاشة فان البيانات تظهر على آخر سطر في الشاشة . أما اذا كانت قيمة رقم السطر صفر ZERO أو غير معرفة نهائياً فان البيانات تظهر في المكان التالي مباشرة لموضع الـ CURSOR حين تنفيذ التعليمة . ورقم السطر يمكن زيادته للانتقال الى سطر لاحق وهكذا ... فاذا وصل التسلسل الى آخر سطر في الشاشة ينتقل بعد ذلك الى أول سطر فيها بعد مسحها ERASE آلياً .

وعبـارة POSITION تـدل على مـكـان وضـع الـرمـز في السطر الواحد على الشاشة (حـيـث أن عـدد الـرمـوز في الـسـطـر يصل الى ٨٠ رمزاً عادة*، وفي بعض الشاشات قد يـكـون أقل من ذلك). والقيمة التي تلي كلمة POSITION يجب أن تكون قيمة عددية صـحـيـحـة كـمـا هـي الحـال في رقم السطر وضمن عدد الأماكن المتاحة على الشاشة. و يـبـدأ الـسـطـر من جهة اليسار بالموضع رقم (١) و ينتهي الى جهة اليمين بالموضع رقم (٨٠) وإذا لـم تذكر القيمة فان الحاسب يفترض وجود القيمة (١). وإذا تم ذكر قيمة الـعدد (صفر) يتم عرض البيانات في الموضع التالي لوجود CURSOR، و ينتهي العرض عند آخر رمز مراد عرضه.

عبارة SIZE تدل على حجم البيانات المعروضة في الحقل أو الثابت الذي يلي كلمة DISPLAY. والقيمة التي تلي كلمة SIZE تحدد هذا الحجم.

عـبارة BEEP تعني عند وصفها في تعليمة DISPLAY إظهار صوت الإشارة الخاصة بالشاشة عند عرض أي رمز من البيانات المطلوب عرضها.

أما وجود عبارة ERASE في تعليمة العرض فيدل على طلب المبرمج مسح أو تنظيف الـشـاشـة كـلها قبل عرض البيانات الواردة في تعليمة DISPLAY. وفي حال عدم ذكر هذه العبارة يتم العرض على الشاشة دون عملية تنظيف لها سابقة للعرض.

أما عبارة LOW أو HIGH فهما كلمتان وجود أحدهما يدل على نوع الحروف المراد اظـهـار فـكـلمة LOW تعني LOW intensity وهي تشكل الحروف العادية على الشاشة أمـا كـلمة HIGH فتعني HIGH intensity و يقصد بها الرموز والحروف ذات الاضاءة الأشـد عـند الإظهار على الشاشة، وتستخدم هذه العبارة لإظهار الحروف في شكلين من شـدة الاضـاءة لأجـل لـفـت نـظـر الـعامل أو مستخدم الشاشة، وتركيز اهتمامه في نوع البيانات المعروضة.

(*) يمكن في بعض شاشات IBM أن تصل سعة السطر الواحد الى ١٣٢ رمزاً.

وتقبل التعليمة DISPLAY إظهار حقل أو ثابت واحد أو أكثر وتُكرر أسماء الحقول أو الثوابت المراد اظهارها مع العبارات الخاصة بمكان وشكل هذه البيانات المطلوب اظهارها على الشاشة، وهذا ما يُقصد بوجود (...) في نهاية الشكل العام للتعليمة.

٤ ـ ٩ تعليمة قبول أو استلام البيانات ACCEPT

إن تعليمة قبول البيانات من خارج البرنامج الى الذاكرة تقوم مقام تعليمة READ للبيانات ولكن ليس من ملف خاص بل من وحدة إدخال بطيئة عادة (سابقا كانت وحدة شبيهة بالآلة الكاتبة تسمى I/O TYPEWRITER) وكان لا ينصح في استخدامها من أجل إدخال حجم البيانات الكبير وشكلها العام كما يلي :

ACCEPT identifier [FROM mnemonic - name]

وكما في تعليمة DISPLAY فأن تسمية رمز وحدة الادخال اختياري وإذا لم يذكر فأن عملية قبول البيانات تتم عن طريق قارىء البطاقات CARD READER المرتبط بالحاسب وإذا ذكرت عبارة FROM فانها تدل على وحدة الإدخال المرغوبة.

و يوجد هناك شكل آخر لتعليمة استلام البيانات في لغة الكوبول COBOL 74 تستخدم للحصول على التاريخ الجاري ليوم التشغيل أو يوم التشغيل أو الوقت عند تشغيل البرنامج. وشكل هذه التعليمة هو :

```
                              { DATE }
ACCEPT identifier FROM        { DAY  }
                              { TIME }
```

وعبارة DATE تعطي التاريخ على أساس ٦ أرقام والرقمان في أقصى اليسار يدلان على السنة الميلادية ثم رقم الشهر ثم رقم اليوم أي أن تاريخ March, 16 1979 يكون على الشكل 790316.

أما اذا اختيرت كلمة DAY فيتوفر لدى المبرمج عدد الأيام منذ اليوم الأول في العام لغاية يوم تشغيل البرنامج (السنة تحتوي على ٣٦٦ يوم). فمثلا يوم March 16, 1979 يظهر على النحو : 79075 بينما يوم March 16, 1980 فيظهر على النحو : 80076 على اعتبار سنة ١٩٨٠ سنة كبيسة.

أما عبارة TIME فهي تعطي الوقت بالساعات والدقائق والثواني فاذا كانت الساعة ١٠ و ١٥ دقيقة صباحاً يظهر في الحقل المستلم للوقت : 10150000 أما الوقت الساعة ١٠ و ١٥ دقيقة مساء فيكون : 22150000 والرقمان الى أقصى اليمين يمثلان الأجزاء المئوية للثانية.

أمثلة :

1) ACCEPT NAME - 1 FROM CONSOLE.

2) ACCEPT STATE - NO.

3) ACCEPT DATE - FLD FROM DATE.

4) ACCEPT DAY - NUM FROM DAY.

5) ACCEPT TIME - FLD FROM TIME.

و بعد تطور صناعة الحاسبات وإدخال الشاشات الضوئية في حقل إدخال البيانات والتعامل المباشر أدى الى تعديل في شكل هذه التعليمات المتبناة من قبل هيئة الـ

(ANSI) وظهرت تعليمة ACCEPT لتلائم هذا التطور فكان شكلها العام* على النحو التالي :

```
ACCEPT identifier - 1 [ , UNIT { identifier - 2 | literal - 1 } ] [ , LINE
    { identifier - 3 | literal - 2 } ] [ , POSITION { identifier - 4 | literal - 3 } ]
    [ , SIZE { identifier - 5 | literal - 4 } ] [ , PROMPT [ literal - 5 ] ]
    [ , ECHO ] [ , CONVERT ] [ , TAB ] [ , NO BEEP ]
    [ , ERASE ] [ , OFF ] [ { , LOW | , HIGH } ]
    [ , identifier - 6 [ , UNIT { identifier - 7 | , literal - 6 } ] [ , LINE
    { identifier - 8 | literal - 7 } ] [ , POSITION { identifier - 9 | literal - 8 } ]
    [ , SIZE { identifier - 10 | literal - 9 } ] [ PROMPT [ literal - 10 ] ]
```

(*) TI990 COBOL

```
[ , ECHO ] [ , CONVERT ] [ , TAB ] [ , NO BEEP ]

                            [ [ , LOW  ] ]
[ , ERASE ] [ , OFF ]       [ {        } ]      ....
                            [ [ , HIGH ] ]

[ ON   EXCEPTION   identifier - 11, imperative statement ]
```

وتعليمة ACCEPT هذه تستخدم لنقل البيانات من خلال الشاشة الضوئية الى الحاسب لتتيح استخدامها ومعالجتها بوساطة البرنامج وتخزن هذه البيانات في المتغير المسمى بالحقل identifier - 1 و identifier - 6 ... وغيره.

وعبارة UNIT تعني تحديد رقم الشاشة التي يتم بموجبها إدخال المعلومات وقبولها الى داخل البرنامج، وإذا لم تحدد فتكون هي الشاشة الذي يُشغّل من خلالها البرنامج.

أما عبارات LINE و POSITION و SIZE فهي تحدد رقم السطر والموضع وحجم البيانات المدخلة في تعليمة القبول و ينطبق عليها ما ورد في شرح تعليمة DISPLAY.

أما عبارة PROMPT فتفيد في تحديد حجم البيانات المراد قبولها وهي تُظهر، عند استخدامها، على الشاشة إشارة (*) أو (-) على عدد من المرات يساوي الحقل الموصف في قسم البيانات والمخصص لاستلام البيانات المطلوبة. وعند إضافة الثابت literal - 5 أو literal - 10 بجانبها عندئذ يمكن تحديد شكل آخر غير إشارة (*) للظهور قبل إدخال البيانات ويجب أن يكون أو تكون هذه الاشارة أو الرمز غير رقمي.

عبارة ECHO عند استخدمها تتيح إعادة عرض البيانات المقبولة للتأكد من صحة استلامها على الوجه المطلوب.

أما إذا كانت البيانات المراد ادخالها رقمية وتحتوي على فاصلة عشرية عندئذ يجب استخدام عبارة CONVERT مع التعليمة من أجل تخصيص الأماكن العشرية

والـصحـيحـة وتخـزيـنها في الوضع الصحيح أثناء عملية قبول البيانات. وفي حال عدم ذكرها في التعليمة يعتبر الرقم المدخل كله عدداً صحيحاً.

كلمة : TAB تمكن مدخل البيانات من التأكد من البيانات التي يدخلها وتتيح له قراءتها ثم ضغط الزر (الملمس) الخاص RETURN أو NEW - LINE لقبول المعلومات المدخلة، وعدم ذكر هذه الكلمة يعني قبول البيانات المدخلة آلياً دون ضغط ملمس RETURN في نهاية الادخال للحقل المطلوب.

ووضع عبارة NO BEEP تسمح باخفاء صوت BEEP الذي تظهره الشاشة عند إدخال أي رمز. واغفال هذه العبارة يؤدي الى ظهور الصوت BEEP (بيب).

أما كلمة ERASE فتقوم عند استخدامها بمسح وتنظيف الشاشة قبل عملية ادخال الحقل (البيانات) المطلوبة بتعليمة ACCEPT.

أما كلمة OFF تسمح بادخال البيانات على الشاشة مع عدم ظهورها أثناء إدخالها من لوحة المفاتيح والملامس وتستخدم هذه الكلمة لادخال كلمات السر (PASS WORD) في البرنامج.

أما كلمتا LOW و HIGH فهي تستخدم كما ورد في تعليمة DISPLAY.

أما وجود عبارة ON EXCEPTION مع تعليمة ACCEPT يتيح للمبرمج استخدام المفاتيح الخاصة المسماة FUNCTION KEYS وتختلف هذه عدداً وكيفاً باختلاف الحاسب المستخدم. إلا أن الشكل العام لهذه المفاتيح هو استخدامها كأداة مقارنة وشرط لقبول عمل ما أو الانتقال الى عمل آخر.

أمثلة :

1) ACCEPT FUNC - 1, FUNC - 2.

2) ACCEPT ANSWER - 1 LINE 1 POSITION MI PROMPT

```
ECHO CONVERT.
```

3) ACCEPT NEXT - L POSITION 0 PROMPT ECHO.

4) ACCEPT YEAR LINE 10 POSITION 10 MONTH LINE 11 POSITION 10.

5) ACCEPT ANSWER - 2 ON EXCEPTION STATUS - WD, GO TO DSP - WRD.

٤ ـ ١٠ تعليمة «أفحص» INSPECT

تستخدم تعليمة «افحص» INSPECT لأجل تدقيق أي حقل واستبدال رمز معين برمز آخر في هذا الحقل. وتفيد في شكل خاص أثناء عملية التحقق من البيانات. ويمكن أن تقوم هذه التعليمة بعد (احصاء) رمز معين ضمن الحقل ومعرفة عدد تكراره. ويمكن تلخيص التطبيقات الخاصة بهذه التعليمة في نقطتين :

١ ـ عد أو احصاء عدد مرات تكرار رمز معين Character في حقل ما identifier.

٢ ـ استبدال رمز معين ومكرر ضمن حقل ما برمز آخر.

ولهذه التعليمة شكلان عامان :

الشكل الأول :

```
INSPECT identifier - 1 TALLYING

  {                        { ALL        }   ( identifier - 3 )
  {  identifier - 2 FOR    { LEADING    }   (                )
  {                        { CHARACTERS }   ( literal - 1    )

  [ { BEFORE }              { identifier - 4 } ] }
  [ {        }   INITIAL    {                } ] }
  [ { AFTER  }              { literal - 2    } ] }
```

أمثلة :

1) INSPECT FIELD - 1 TALLYING CNTR - 1 FOR ALL SPACES.

2) INSPECT FIELD - 2 TALLYING CNTR - 2 FOR CHARACTERS BEFORE INITIAL SPACE.

3) INSPECT FLD - 3 TALLYING CNTR - 3 FOR LEADING ZEROS.

4) INSPECT FLD - 4 TALLYING CNTR - 4 FOR ALL "A" AFTER INITIAL "C".

يكون محتوى العدادات بعد التنفيذ :	إذا كانت قيمة الحقل Identifier Value :
CNTR - 1 = 4	FLD - 1 =١□□□ 51 □
CNTR - 2 = 5	FLD - 2 = ϕ1725 □
CNTR - 3 = 2	FLD - 3 = ϕϕ76ϕϕ
CNTR - 4 = 2	FLD - 4 = A B X C A A

وفي هذه الأمثلة فان الحقول FLD - 1 الى FLD - 4 هي اماكن معرفة في قسم البيانات بوساطة عبارة PIC. وكذلك الحقول CNTR - 1 الى CNTR - 4 هي اماكن موصوفة في نفس القسم بوساطة عبارة PIC لأجل حفظ نتيجة عدد الرموز المطلوب (فحصها) أو عدها بوساطة التعليمة INSPECT TALLYING.

و يقصد أيضا بكلمة ALL جميع الرموز وكلمة LEADING الرموز الموجودة في مقدمة الحقل من جهة اليسار.

الشكل الثاني

وتعليمة «أفحص» يمكن أن تقوم بوظيفتها الثانية وهي استبدال الرموز في الحقل عند إضافة جملة REPLACING بدلاً من جملةTALLYING و يصبح شكلها كما يلي :

```
INSPECT identifier - 1 REPLACING

    { ALL     }          { identifier - 2 }
    { LEADING }          { literal - 1    }
    { FIRST   }

              CHARACTERS
```



حيث أن معنى كلمة ALL و LEADING يأخذ نفس المعنى في الشكل الأ ول وكلمة FIRST يقصد بها أول رمز. وهذه التعليمة تقوم باستبدال الرمز المعرّف في الحقل identifier - 2 و literal - 1 بالرمز المعرَّف في الحقل identifier - 3 أو الثابت literal - 2 .

أمثلة :

1) INSPECT FLD - E REPLACING ALL "1" BY "2".

2) INSPECT FLD - E REPLACING LEADING "1" BY "2".

3) INSPECT FLD - E REPLACING CHARACTERS BY "3"
   BEFORE INITIAL "2".

4) INSPECT FLD - E REPLACING FIRST "X" BY "Y".

	إذا كانت قيمة الحقل FLD-E قبل التنفيذ :	تكون قيمة الحقل FLD-E بعد التنفيذ:
1)	= 112111	= 222222
2)	= 112111	= 222111

3) = 112111 = 332111

4) = ABCXYZ = ABCYYZ

الشكل الثالث :

يمكن جمع وظيفتي تعليمة «أفحص» وهما وظيفة عد (احصاء) الرموز ووظيفة استبدالها في تعليمة واحدة INSPECT ويمكن أن تأخذ الشكل العام التالي :

```
INSPECT  identifier - 1     TALLYING     identifier - 2

     FOR  { ALL       }  { identifier - 3 } }
          { LEADING   }  { literal - 1    } }
                   CHARACTERS

     [ { BEFORE }   INITIAL   { literal - 2    } ]
     [ { AFTER  }             { identifier - 4 } ]

REPLACING
{ { ALL     }  { identifier - 5 } }      { identifier - 6 }
{ { LEADING }  {                } }  BY  {                }
{ { FIRST   }  { literal - 3    } }      { literal - 4    }
                   CHARACTERS

     [ { BEFORE }   INITIAL   { identifier - 7 } ]
     [ { AFTER  }             { literal - 5    } ]
```

مثال :

```
MOVE ZERO TO CNT - X.
INSPECT FLD - 1 TALLYING CNT - X FOR LEADING "B"
    REPLACING ALL "B" BY "R".
```

فإذا كان الحقل FLD - 1 في المثال السابق يحتوي على البيانات التالية قبل تنفيذ تعليمة INSPECT :

FLD - 1 = BBBBABCD

وكذلك يكون CNT - X مساوياً للصفر بوساطة تعليمة MOVE فان هذين الحقلين يصبحان بعد التنفيذ على النحو التالي :

FLD - 1 = RRRRARCD

CNT - X = ∅4

حيث أنهما يأخذان الصورة التالية في التوصيف في قسم البيانات :

```
FLD - 1 PIC X (8).
CNT - X PIC 99.
```

وتجدر الاشارة هنا الى أن تعليمة INSPECT التي أدخلت الى لغة كوبول* في التحسينات الأخيرة (عام ١٩٧٤) تعادل الى حد قريب تعليمة EXAMINE التي كانت موجودة في نسخة لغة كوبول قبل عام ١٩٧٤م وكانت تعليمة EXAMINE تأخذ الشكلين العامين التالين :

```
                                      ⎧ UNTIL FIRST ⎫
1) EXAMINE  identifier  TALLYING      ⎨ ALL         ⎬  literal - 1
                                      ⎩ LEADING     ⎭
              [REPLACING BY  literal - 2]
```

---

* STRUCTURED COBOL PROGRAMMING, stern / stern, John Wiley , U.S.A, 1980, 3ed. P.P. 364

حيث يتم تخزين عدد الرموز المعدودة (المحصاة) في الحقل TALLY الذي يقوم نظام التشغيل (أو مترجم الكوبول) بتعيينه وتعريفه و يستطيع المبرمج معالجته بهذا الاسم.
أما الشكل الثاني المخصص لاستبدال الرموز فهو على النحو التالي :

```
2) EXAMINE   identifier   REPLACING

   { ALL              }
   { LEADING          }
   { FIRST            }  literal - 1 BY literal - 2
   { UNTIL FIRST      }
```

مثال :

```
EXAMINE NAME1 REPLACING ALL "0" BY SPACE.
```

٤ - ١١ أسئلة وتمارين

سؤال ١ - بين الصحيح من العبارات التالية باشارة (✓) والخطأ منها باشارة (×):

□ ١ - تستخدم عبارة التطابق CORRESPONDING مع تعليمة النقل MOVE فقط.

□ ٢ - يمكن استخدام عبارة ROUNDED وعبارة ON SIZE ERROR مع أي من التعليمات الحسابية.

□ ٣ - يجب أن تحتوي تعليمة COMPUTE عبارة ROUNDED لإعطاء القيمة المحسوبة المقربة.

□ ٤ ـ عند استخدام تعليمة ACCEPT للحصول على اليوم DAY يمكن معرفة اسم اليوم في السنة مباشرة.

□ ٥ ـ عند استخدام تعليمة ACCEPT للحصول على الوقت TIME يتطلب حجز حقل طوله ستة خانات (digits) للحصول عليه.

□ ٦ ـ استخدام تعليمة WRITE مع تعليمة FROM يغني عن استعمال تعليمة MOVE قبلها لأجل نقل البيانات المراد كتابتها.

□ ٧ ـ تأخذ تعليمة INSPECT شكلين مختلفين.

سؤال ٢ ـ اذا كانت لدينا تعليمة IF المتداخلة على النحو التالي :

الشكل الأول

```
IF     M-STATUS  = 'S'
        PERFORM SINGLE - RTN
ELSE
    IF     M-STATUS = 'M'
        PERFORM MARRIED - RTN
    ELSE
        PERFORM ERROR - RTN.
```

وكان لدينا أيضا عدة تعليمات IF وفق الشكل التالي :

الشكل الثاني

```
IF     M-STATUS  = 'M' PERFORM MARRIED - RTN.
IF     M-STATUS  = 'S' PERFORM SINGLE - RTN.
```

```
IF     M-STATUS NOT  = 'S'
       AND M-STATUS NOT 'M' PERFORM ERROR - RTN.
```

والمطلوب إيضاح ما يلي :

١ ـ هل الشكل الأ ول يعادل الشكل الثاني و يعطي نفس النتيجة .

٢ ـ في حال كون هذين الشكلين متكافئين فايهما أفضل من ناحية البرمجة وحسن تنفيذ البرنامج .

٣ ـ هل تكون نتيجة التنفيذ واحدة لو استبدلنا العامل AND بالعامل OR في الشكل الثاني .

سؤال ٣ ـ لدينا الجزء التالي من برنامج كوبول والذي وضع لحساب عمر الفرد الموظف من واقع سجله RECORD المخزن على وسيط مغناطيسي ، ويحتوي هذا السجل على تاريخ ميلاده EMP - BRITH - DATE ، أما تاريخ التنفيذ للبرنامج فيتم الحصول عليه بوساطة تعليمة ACCEPT مع عبارة DATE لأجل حساب الفترة الزمنية السابقة ومعرفة العمر الصحيح .

وعلى الرغم من أن ترميز هذا البرنامج صحيح من الناحية الشكلية إلا أنه لا يعطي النتيجة المطلوبة عند تنفيذه لأجل معرفة العمر. فالمطلوب معرفة أسباب ذلك وتصحيح هذا البرنامج على ضوء قواعد استخدام تعليمة ACCEPT وطريق حساب العمر.

```
ϕ1     EMP - REC.
   ϕ5     EMP - BIRTH - DATE.
       1ϕ     BIRTH - MONTH       PIC 99.
       1ϕ     BIRTH - YEAR       PIC 99.
        .
         .
          .
```

```
ϕ1     EMPL - AGE                PIC 99.
          .
          .
          .
ϕ1 DATE - WORK - AREA.
    ϕ5     TODAYS - MONTH            PIC 99.
    ϕ5     TODAYS DAY            PIC 99.
    ϕ5     TODAYS - YEAR            PIC 99.
          .
          .
          .
PROCEDURE DIVISION.
    ACCEPT DATE - WORK - AREA FROM DATE.
          .
          .
          .
    COMPUTE EMPL - AGE  =  TODAYS - YEAR
            —     BIRTH - YEAR + TODAYS - MONTH
            —     BIRTH - MONTH.
```

سؤال ٤ ـ إذا كان لدينا حقل اسمه CITY - CODE وطوله رمز واحد (بايت) والممكن أن يحتوي هذا الحقل على أحد الرموز التالية :

S, M, D, W

والمطلوب توصيف هذا الحقل مستخدماً اسم الشرط Condition - name في قسم البيانات.

تمرين ١ :

أكتب برنامجاً بلغة الكوبول يقبل عن طريق الشاشة TERMINAL رقم الموظف، واسمه، وراتبه وبدل النقل، ويسجل هذه المعلومات على ملف متسلسل SERIAL

موجود على القرص الممغنط واسمه SALFILE وشكل السجل فيه كالتالي :

	FIELD NAME	PICTURE
رقم الموظف	EM - NO	9 (4)
اسمه	EM - NAME	X (2ø)
راتبه	EM - SAL	9 (5) V 99
بدل النقل	EM - TRANS	9 (3) V 99
غير مستعمل	BLANK	X (4)

أظهر الرسائل المناسبة عند إدخال كل حقل.
وفي نهاية العمل أدخل الرمز المناسب للانتهاء.

* أعمل تحقق للحقول السابقة بحيث أن :

رقم الموظف وراتبه و بدل النقل يجب أن تكون رقمية وأكبر من الصفر أما اسم الموظف فيجب ألا يكون فراغاً.

تمرين ٢ :

احدى الشركات الخاصة تريد عمل صندوق لمساعدة العمال، وذلك باستقطاع مبلغ معين من الأجر الشهري لكل عامل وفق القواعد التالية :

أ. اذا كان أجر العامل ١٠٠٠ ريال وما دون يقتطع عليه ١٪ من الأجر الشهري لصالح الصندوق.

ب. اذا كان أجر العامل أكبر من ١٠٠٠ ريال وأقل من ٢٠٠١ ريال يقتطع عليه :

١٪ عن الألف الأولى من أجره الشهري
٣٪ عن المبلغ المتبقي من الأجر الشهري

جـ. إذا كان أجر العام أكبر من ٢٠٠٠ ريال يقتطع عليه :
٪١ عن الألف الأولى من الأجر الشهري
٪٣ عن الألف الثانية من الأجر الشهري
٪٥ عن باقي الأجر الشهري (ما فوق ٢٠٠٠ ريال)

المطلوب :

اكتب برنامجاً يحسب صافي الأجر لكل عامل علماً بأنه يحسم عليه ٥٪ من الأجر الشهري (قبل حسم الصندوق) كتأمينات اجتماعية.
علماً بأن الملف موجود على قرص ممغنط اسمه PAYFILE وأن توصيف السجل فيه كالتالي :

	FIELD NAME	PICTURE
رقم الموظف	EMP - NO	9 (4)
اسمه	EMP - NAME	X (2ϕ)
الأجر الشهري	EMP - SAL	9 (5) V 99
بدل النقل	EMP - TRANS	9 (3) V 99
	BLANK	X (4)

واطبع رقم الموظف، اسم الموظف، الأجر الشهري، بدل النقل، تبرع الصندوق، التأمينات الاجتماعية، صافي الراتب، على الطابعة من أجل كل عامل - أنظر شكل المخرجات صفحة ٦٠٢ والنتائج صفحة ٥٩٩.

١ - اطبع رقم الصفحة والتاريخ مع العنوان في راس كل صفحة
٢ - اترك سطر فراغ بين كل سطري طباعة
٣ - لا تطبع في الصفحة الواحدة أكثر من ٢٥ سطراً

أمثلة على طريقة حساب صافي الأجر الشهري :

مثال ١ - بفرض أن أحد العمل أجره الشهري ٢٥٠٠ ريال و بدل النقل ٣٠٠ ريال فيكون

١ - التأمينات

$$\frac{٢٥٠٠ \times ٥}{١٠٠} = ١٢٥ \text{ ريالا}$$

٢ - التبرع

١٠٠٠ × ١٪ = ١٠ ريالات

١٠٠٠ × ٣٪ = ٣٠ ريالا

٥٠٠ × ٥٪ = ٢٥ ريالا

المجموع ٦٥ ريالا

٣ - الصافي = (٢٥٠٠ + ٣٠٠) - (١٢٥ + ٦٥) = ٢٦١٠

مثال ٢ - الأجر الشهري ١٢٠٠ ريال و بدل النقل ٣٠٠ ريال يكون :

١ - التأمينات $\frac{١٢٠٠ \times ٥}{١٠٠}$ = ٦٠ ريالا

٢ - التبرع ١٠٠٠ × ١٪ = ١٠ ريالات

٢٠٠ × ٣٪ = ٦ ريالات

المجموع ١٦ ريالا

٣ - الصافي : (١٢٠٠ + ٣٠٠) - (١٦ + ٦٠) = ١٤٢٤ ريالا

تمرين ٣ - ملف من البطاقات المثقبة اسمه STDFILE يحتوي على اسم الدارس ودرجته النهائية بمعدل بطاقة لكل دارس :

	FIELD NAME	PICTURE
اسم الدارس	STD - NAME	X (25)
الدرجة النهائية	STD - MARK	9 (3)
———	FILLER	X (52)

اكتب برنامجاً بلغة الكوبول يقرأ هذا الملف و يطبع محتواه على الطابعة مع ملاحظة القواعد التالية :

- اذا كانت الدرجة STD- MARK أقل من ٦٠ أطبع كلمة 'FAIL' راسب أمام الدرجة
- اذا كانت الدرجة من ٦٠ إلى ٦٩ أطبع كلمة 'PASS' ناجح
- اذا كانت الدرجة من ٧٠ إلى ٧٩ أطبع كلمة 'GOOD' جيد
- اذا كانت الدرجة من ٨٠ إلى ٨٩ أطبع كلمة 'V.GOOD' جيد جداً
- اذا كانت الدرجة تسعين فما فوق اطبع كلمة 'EXCELLENT' ممتاز

مع ملاحظة أن الدرجة العظمى هي ١٠٠ .

شكل سجل الاخراج :

xxxxxxxxxxxxxxxxxxxxxxxxx	999	xxxxxxxxx
للاسم	للدرجة	للتقدير

الفصل الخامس

أقسام برنامج كوبول الأخرى والعبارات المتقدمة

٥ - ١ لمحة عامة :

يفيد هذا الفصل بالإضافة الى المبتدئين في برمجة كوبول، المبرمجين الذين يودون التقدم أكثر في لغة كوبول حيث أنه يزود القارىء بتفاصيل كثيرة عن بقية أقسام البرنامج (قسم البيانات) و (قسم المحيط). وخاصة فيما يتعلق باظهار أو اخراج البيانات والرموز المستخدمة لذلك، وبقية توصيف البيانات مثل عبارة USAGE وأسماء الشرط وعبارات إعادة التوصيف والتسمية REDEFINES, RENAMES علاوة على ذكر تعليمة النسخ COPY، وفي نهاية الفصل ترد البرامج الفرعية والرئيسية وكيفية الربط بينهما وفصل الربط LINKAGE SECTION.

٥ - ٢ رموز الاظهار (الاخراج)* EDITING CHARACTERS

تعتبر الرموز التالية من الرموز المساعدة على إخراج الحقل (عرض البيانات) على شكل معين :

B, 0, +, -, Z, *, $, ., / , CR, DB

وهذه الرموز تشير الى عملية الاظهار المطلوبة في الحقل. ويمكن تصنيف هذه الرموز الى أربعة أصناف :

١ - عدم إظهار بسيط (Simple Supression) : وهي الرموز التي تسبب عدم ظهور

(*) Sordillo Donald, **THE PROGRAMMERS ANSI COBOL REFERENCE MANUAL**, Prentice - Hall, 1978, PP. 193-203.

بعض البيانات في الحقل.

٢ ـ عدم إظهار واستبدال Suppression with Replacement : وهي الرموز التي تسبب عدم ظهور بعض البيانات و بنفس الوقت تمكن من استبدالها بشكل آخر.

٣ ـ الاستبدال الفائض Floating Replacement : وهي الرموز التي تسبب إدخال رمز أو رموز في الحقل عند إظهار بياناته في حالة معينة.

٤ ـ الإدخال (الاقحام) الثابت Fixed Insertion : وهي الرموز التي تؤدي الى إدخال رمز أو رموز وتبقى ثابتة عند إظهار البيانات.

٥ ـ ٢ ـ ١ الطمس البسيط Simple Suppression

يؤدي الرمز Z الى طمس جميع الأصفار التي تظهر في الحقل الرقمي من جهة ا ليسار (أي الأصفار غير المعنوية) وكذلك يمكن طمس قيمة الحقل المراد إظهاره اذا كانت قيمته صفراً وكان حقل الاظهار Edited field موصوفا بالرمز Z.

واذا كان الحقل المراد إظهاره يحتوي على رقم عشري وتم وصف الجزء الصحيح منه بالرمز Z فإن الأصفار التي تقع على جهة اليسار من الجزء الصحيح فقط تطمس. أما اذا كان الجزء العشري من الحقل موصوفاً بالرمز Z فيجب أن تكون جميع الأ رقام التالية من جهة اليسار موصوفة به كذلك. ولا يتم طمس الأصفار في هذه الحالة إلا على يسار الفاصلة العشرية. واذا كان الرقم بكامله مساو ياً للصفر فان القيمة المظهرة تكون فراغاً حتى بدون ظهور الفاصلة العشرية. لأن أي رمز ثابت Fixed insertion مقحم في شكل الحقل يعامل مثل الرمز Z.

والجدول التالي* يبين الأشكال المختلفة لاظهار حقل رقمي :

(*) Sordillo P. 195.

Data	PICTURE						
	9999	Z999	ZZ99	ZZZ9	ZZZZ	ZZ.ZZ	Z,ZZZ
0	0000	000	00	0			
1.01	0001	001	01	1	1	1.01	1
217.	0217	217	217	217	217	17.00	217
1010.20	1010	1010	1010	1010	1010	10.20	1, 010
.01	0000	000	00	0		.01	

٥ ـ ٢ ـ ٢ الطمس مع الاستبدال

SUPPRESSION WITH REPLACEMENT

يتم طمس الأصفار من جهة يسار القيمة مع استبدالها باشارة النجمة * بدلاً من الفراغ بوساطة الرمز المستخدم للإظهار «*» والمسمى Asterisk . وعند استخدام النجمة كرمز لتوصيف حقل الإظهار يضاف إلى جانب الحقل الرقمي المراد اظهاره من جهة اليسار نجوماً بدلاً من الأصفار غير المعنوية. وتستخدم هذه الطريقة في إظهار الحقول الرقمية بشكل خاص في طباعة الشيكات المصرفية أو تدوين القيمة المالية تلافياً لامكانية زيادة القيمة والتزوير.

ويجب توصيف الحقل بعدة نجوم متصلة قد تفصل بينها رموز ثابتة (Fixed Insertion Characters) .

ولا يمكن استخدام رمز النجمة اذا كان الحقل المراد إظهاره موصوفاً بعبارة BLANK WHEN ZERO .

الجدول التالي* يوضح كيفية استخدام توصيف رمز النجمة في إظهار الحقول الرقمية :

PICTURE					
Data	*999	**99	***9	**,**	*,***.**
0	*000	**00	***0	**.**	*****.**[١]
13	*013	**13	**13	13.00	13.00
123	*123	*123	*123	23.00	**123.00
2010	2010	2010	2010	10.00	2,010.00
1010.2	1010	1010	1010	10.20	1,010.20
.01	*000	**00	***0	**.01	*****.01[١]

٥ - ٢ - ٣ الاستبدال العائم FLOATING REPLACEMENT

يستخدم لأجل الاستبدال العائم في إظهار الحقل ثلاثة رموز وهي :

علامة العملة المتداولة Currency Symbol

اشارة الموجب +

اشارة السالب —

وتُمثل علامة النقد برمز واحد مثل ($) أو (£) أو أي رمز آخر حسب نوع العملة (على أن يكون هذا الرمز متوفر على وحدات الإدخال والإخراج) . و يتم تحديده في عبارة CURRENCY SIGN IS في الفقرة المسماة للأسماء والتعريفات الخاصة ؛ من قسم المحيطات SPECIAL - NAMES .

(*) Sordillo P. 196

(١) يلاحظ أن مكان الفاصلة العادية ظهر نجمة آما الفاصلة العشرية فقد ظهرت كما هي.

كـمـا يمـكـن أن تـكـون عـلامة العملة المتداولة رمز اقحام ثابت. وعند استخدامها كرمز عائم تظهر العلامة الى جانب آخر رقم من جهة اليسار و يكون الجزء التالي لها من الحقل فراغاً (أيضاً من جهة اليسار).

والجدول التالي* يوضح كيفية استخدام هذا الرمز في الإظهار:

Data	PICTURE			
	$$99	$$$$9	$$$$$.99	$$,$$$.$$
0	$00	$0	$.00	
.02	$00	$0	$.02	$.02
12	$12	$12	$12.00	$12.00
12.34	$12	$12	$12.34	$12.34
1234	$234	$1234	$1234.00	$1,234.00
1234.56	$234	$1234	$1234.56	$1,234.56

أمـا بـالـنـسبة للاشارتين الجبريتين السالب والموجب فانهما يمكن أن يستخدما إما كـرمـوز ثابتة أو عائمة. واذا استخدم الرمز «علامة السالب» فإن إشارة السالب سوف تظهر في الحقل عند كون قيمة الحقل الأصلي سالبة أما اذا كانت القيمة موجبة أو صفرا فـان الإشـارة لا تـظـهـر. أمـا إذا اسـتخدمت الإشارة الموجبة كرمز فان الإشارة الجبرية للقيمة سوف تظهر سواء كانت القيمة سالبة أم موجبة.

والشكل التالي** يوضح كيفية استخدام هذين الرمزين في إظهار البيانات الرقمية.

(*) Sordillo P. 196

(**) Sordillo P. 197

Data	PICTURE							
	+9999	-9999	++++9	----9	+++++	-----	+++++.++	--,---.--
0	+0000	0000	+0	0				
12	+0012	0012	+12	12	+12	12	+12.00	12.00
-12	-0012	-0012	-12	-12	-12	-12	-12.00	-12.00
123	+0123	0123	+123	123	+123	123	+123.00	123.00
-123	-0123	-0123	-123	-123	-123	-123	-123.00	-123.00
-123	-0123	-0123	-123	-123	-123	-123	-123.00	-123.00
1234	+1234	1234	+1234	1234	+1234	1234	+1234.00	1,234.00
-1234	-1234	-1234	-1234	-1234	-1234	-1234	-1234.00	-1,234.00
12.34	+0012	0012	+12	12	+12	12	+12.34	12.34
-12.34	-0012	-0012	-12	-12	-12	-12	-12.34	-12.34

٥ ـ ٢ ـ ٤ الاقحام الثابت FIXED INSERTION

إن الرموز المستخدمة في الاقحام الثابت عند اظهار حقل ما هي : الفاصلة ، الصفر 0 و / و B (تعني فراغ Blank) و CR (تعني دائن Credit) و DB (تعني مدين Debit). وكذلك وجود رمز واحد فقط في توصيف حقل الاظهار من الرموز التالية ــ و + واشارة العملة المتداولة ($).

تُحصى جميع الرموز الثابتة عند حساب حجم الحقل المراد اظهاره وإشارتا CR و DB تأخذ موضعين عند تحديد حجم الحقل. كما يجب الاشارة الى أن الفاصلة (وليس المقصود الفاصلة العشرية بل الفاصلة التي تحدد الألوف في الرقم) لا يمكن أن تكون آخر رمز في توصيف الحقل المُظْهر.

والشكل التالي* يوضح الاستخدامات المختلفة لهذه الرموز في اظهار حقل البيانات :

Data	PICTURE					
	99B99	9,999	99/909	**,***	9B9/900	99,99
0	00 00	0,000	00/000	******	0 0/000	00,00
12	00 12	0,012	00/102	****12	0 1/200	00,12
123	01 23	0,123	01/203	***123	1 2/300	01,23
1234	12 34	1,234	12/304	*1,234	2 3/400	12,34

أما إشارتا الدائن والمدين DB و CR فيجب وضعها على يمين الرموز الموصّفة للحقل المظهر. ولا يمكن استخدامها مع الحقول الموصوفة بالرموز التالية : A X - + أو S كما لا يمكن وضع الاشارتين في نفس التوصيف الواحد. ومن الشكل التالي** يتبين أن إشارة الدائن أو المدين تظهر في الحقل عند وجود القيمة سالبة فقط. وتختفي عند كون القيمة صفر أو موجبة.

Data	PICTURE					
	$9999CR	$9999DB	$$,$$$CR	$$,$$$DB	ZZZZCR	ZZZZDB
0	$0000	$0000				
12	$0012	$0012	$12	$12	12	12
-12	$0012CR	$0012DB	$12CR	$12DB	12CR	12DB
1234	$1234	$1234	$1,234	$1,234	1234	1234
-1234	$1234CR	$1234DB	$1,234CR	$1,234DB	1234CR	1234DB

(*) Sordillo 198

(**) Sordillo 198

هـذا ويمـكـن تـوليـف أكـثـر مـن رمـز في تـوصيف الحقل المظهر. كما يمكن وضع الاشارتين الجـبـريتين على يمين الحقل الموصوف بدلاً من جهة الشمال والجدول التالي* يعطي ملخصاً عن كيفية هذه الاستخدامات المختلفة.

Data	PICTURE							
	+999	-999	999+	999-	+$999	-$999	$∘∘∘∘CR	$ZZZZCR
0	+000	000	000+	000	+$000	$000	$∘∘∘∘	
12	+012	012	012+	012	+$012	$012	$∘∘12	$ 12
-12	-012	-012	012-	012-	-$012	-$012	$∘∘12CR	$ 12CR
123	+123	123	123+	123	+$123	$123	$∘123	$ 123
-123	-123	-123	123-	123-	-$123	-$123	$∘123CR	$123CR

٥ ـ ٢ ـ ٥ الفاصلة العشرية DECIMAL POINT

لا يمـكـن تـصنيف الفاصلة العشرية ضمن أي نوع من أنواع الرموز السابقة لأنها لا تعتبر فقط رمز اقحام ثابت Fixed insertion بل تدل على الجزء العشري والجزء الصحيح مـن الـعـدد. كما لا يمكن أن يظهر في الحقل إلا فاصلة عشرية واحدة (يعبر عنها باللغة الانـكـلـيـزية Point أي النقطة) وكذلك لا يمكن أن تُستخدم مع وجود الرموز التالية في نـفـس الـتـوصيـف للحقل A و X و P و V. وهذه الفاصلة تقوم بتحديد الجزء العشري والصحيح من الحقل الرقمي الحقيقي (Real) ووظيفتها مثل وظيفة الرمز V أو P. وعند الاظـهار تُعد الفاصلة مع حساب حجم الحقل المظهر ولا يمكن أن تكون آخر رمز في رموز توصيف الحقل.

(*) Sordillo P. 199

وتجب الإشارة الى أن عبارة DECIMAL - POINT IS COMMA التي تكتب في فقرة الأسماء الخاصة SPECIAL - NAMES في قسم المحيطيات تمكن المبرمج من استبدال (النقطة العشرية Decimal Point)* بفاصلة عشرية تحت نفس الشروط.

٥ ـ ٣ عبارة اعادة التوصيف REDEFINES

من أجل توفير المساحة المعرّفة في قسم البيانات وتوفير المرونة في استخدام نفس المكان الموصّف في الذاكرة بأكثر من شكل وجدت عبارة REDEFINES والتي تعني اعادة تعريف أو توصيف نفس الحقل (أو المتغير) باسم آخر أو شكل (PICTURE) آخر.

ويمكن أن تأخذ عبارة اعادة التوصيف أي رقم مستوى .Level No من 01 الى 49 وتأتي بعد الحقل المراد توصيفه مباشرة دون أي فاصل أو انقطاع.

Level No. data - name - 1 PICTURE clause

Level No. data - name - 2 REDEFINES data - name - 1 [PIC clause]

(*)الرمز P يستخدم في توصيف الحقل للدلالة على وجود فاصلة عشرية مفترضة في العدد ولا يمكن أن يأتي مع الرمز V ويمكن أن يكون من جهة اليمين أو اليسار في توصيف الحقل الرقمي المراد اظهاره.

مثال :

الرقم 12345 لو كان موصوفاً بالشكل التالي

PPPPP99999

فإنه يظهر على الشكل .0000012345

أما اذا كان موصوفاً بما يلي 99999PPPPP

فانه يظهر على الشكل 1234500000

حيث أن كل P يمثل صفرا عند إظهار الحقل ولا تدخل في حساب الحجم داخل الذاكرة.

مثال :

```
ϕ1   REC - ONE.
     ϕ5   FLD - IN - A   PIC   X (5).
     ϕ5   FLD - IN - B   PIC   99.
ϕ1   REC - TWO REDEFINES REC - ONE   PIC   X (7).
```

أو :

```
ϕ2   UNIT - PRICE   PIC   9 (4).
ϕ2   D - UNIT - PRICE REDEFINES UNIT - PRICE   PIC   99V99.
```

ويجب التـأكيد هنا على أن يكون طول الحقل المعاد توصيفه في نفس الحجم دون زيادة أو نقصان في عدد الرموز (البايت) المكونة له .

كـما أن قواعد لغة كوبول لا تسمح بإضافة عبارة تحديد القيمة VALUE مع عبارة REDEFINES ، لأن ذلك يتعارض منطقياً (اعطاء قيمتين مختلفتين لنفس الحقل) .

كـما تجدر الإشارة هنا الى أنه يمكن إعادة توصيف الحقل الواحد أكثر من مرة . أي يمكن استعمال أكثر من عبارة REDEFINES لنفس الحقل .

٥ - ٤ توصيف الإشارة الجبرية مع العدد SIGNED NUMBERS

يضاف الحرف S على وصف الحقل الرقمي في بدايته من أجل الاشارة الى أن هذا الحقل يحتوي على عدد مع اشارة جبرية والحرف S ليس له أهمية اذا كان الرقم المخزن في الحقل موجبا ولكن أهميته تظهر إذا كان الحقل يحتوي على عدد سالب . أو كان نتيجة عملية حسابية جمع أو طرح . وإذا حُذف الحرف S من التوصيف فان الحقل

يفترض أن يكون موجباً. ولنأخذ الحقول الموصفة فيما يلي :

```
ϕ5   FIELD - A   PIC   S99   VALUE IS - 3ϕ.
ϕ5   FIELD - B   PIC    99   VALUE IS - 25.
```

فإذا كانت نتيجة عملية حسابية سوف تخزن في الحقل FIELD - A فإن ذلك يضمن صحة النتيجة لوجود مكان للاشارة الجبرية. أما إذا كانت نتيجة عملية حسابية ما سوف تتم وتخزن نتيجتها على الحقل FIELD - B فإن العدد الناتج سوف لا يكون صحيحاً في حال كون القيمة الناجمة سالبة لأن الحقل لا يحتوي موضعاً للإشارة. و يُفترض عندئذ الرقم كعدد موجب. فمثلا في حالة جمع العددين A و B ونخزن النتيجة في A فان الرقم الناتج يكون 5- بينما إذا خزنت النتيجة في B فإن النتيجة سوف تكون 5 بدون إظهار الإشارة وهذا غير صحيح جبرياً وحسابياً.

٥ ـ ٤ ـ ١ عبارة الإشارة SIGN CLAUSE

إن عبارة SIGN اختيارية وعند وجودها بجانب الحقل الموصوف في قسم البيانات تعطي الدلالة على تخصيص BYTE بايت للاشارة الجبرية المصاحبة للرقم وتعطي أيضا المبرمج تحديد مكان الاشارة بالنسبة للرقم (من جهة اليمين أو من جهة اليسار) وتستخدم هذه العبارة فقط عندما يكون الحقل الرقمي موصوفاً بعبارة DISPLAY مع كلمة USAGE، وعدم ذكرها بجانب الحقل أي DISPLAY يعني أنها معرفة ما لم يُعرّف بديل آخر.

والشكل العام لعبارة الإشارة كما يلي :

```
             { LEADING  }
[SIGN IS]    {          }   [SEPARATE CHARACTER]
             { TRAILING }
```

حيث أن ظهور كلمة SEPARATE يدل على تخصيص بايت للاشارة أما عدم وجودها فيعني أن الرقم يتضمن الدلالة على اشارته الموجبة والسالبة بدون بايت خاصة بها . أما كلمة LEADING فيقصد بها أن تُعرّف الاشارة قبل الرقم من جهة اليسار (أي على يسار الرقم مثل 915-) أما كلمة TRAILING فيقصد بها تعريف الاشارة على يمين الرقم أو في نهايته (مثل 405-) .

مثال : ليكن الحقل NUMBER معرف على النحو التالي

```
01      NUMBER PIC S999.
```

وإذا كانت قيمته مساوية للعدد 365 بأن يأخذ الأشكال التالية مع استخدام عبارة SIGN .

```
01   NUMBER   PIC   S999   SIGN IS TRAILING.
```

3	6	5

```
01   NUMBER   PIC   S999   SIGN.IS LEADING
```

3	6	5

```
01   NUMBER   PIC   S999   SIGN IS TRAILING SEPARATE.
```

3	6	5	+

رمز الإشارة الموجبة

```
01   NUMBER   PIC   S999   SIGN IS LEADING SEPARATE.
```

+	3	6	5

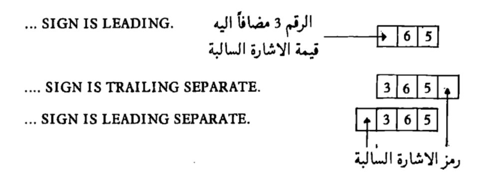
رمز الاشارة الموجبة

أما إذا كانت قيمة الحقل NUMBER مساوية للعدد 365- فيمكن أن يأخذ الأشكال التالية :

... SIGN IS TRAILING.

3	6	+

الرقم 5 مضافاً إليه قيمة الاشارة السالبة



وتجدر الإشارة هنا إلى أن رمز الاشارة أو القيمة المضافة على الرقم للدلالة عليها تختلف باختلاف الحاسب المستخدم.

وعدم ذكر عبارة الإشارة SIGN يعني أن الرقم يحتوي على قيمتها ولا تدخل في بايت خاصة بها.

كما يجب توصيف الرمز S في الصورة (PIC) عند استخدام عبارة SIGN للدلالة على أن الحقل الموصوف معرف برقم مع إشارة جبرية.

٥ ـ ٥ عبارة صيغة التوصيف USAGE

إن وضع البيانات في ذاكرة الحاسب وتخزينها على الوحدات المتصلة به مهم جداً لأنه يحدد طريقة المعالجة المرغوبة للبيانات و يرفع من كفاية الأداء في البرنامج وعبارة USAGE توصف الشكل الذي تكون عليه البيانات داخل الحاسب. ويمكن لهذه العبارة أن توصف الحقول الجزئية Elementary items أو الاجمالية Group items وفي أي جزء من قسم البيانات DATA DIVISION.

والصيغة العامة لاستخدام عبارة USAGE تأخذ الشكل التالي :

```
            { DISPLAY                     }
            { { COMPUTATIONAL - 3 }       }
USAGE IS    { { COMP - 3          }       }
            { { COMPUTATIONAL     }       }
            { { COMP              }       }
```

وهذا الشكل لا يحصر كل الاختيارات الممكن استخدمها مع عبارة USAGE إنما يشمل الأشكال الأساسية والتي تعتبر أكثر استخداماً. ويجب على المبرمج حين استخدام هذه العبارة الرجوع الى مصادر الشركة الصانعة للحاسب لمعرفة الأشكال المتوفرة مع عبارة USAGE. وفي الفقرات التالية شرح عام للاختيارات الأساسية والذي يعتبر شائعاً على أغلب أنواع الحاسبات.

٥ ـ ٥ ـ ١ صيغة DISPLAY

وعبارة USAGE IS DISPLAY تعني أن الحقل الذي عُرِّف من قبل المبرمج في قسم البيانات سوف يمثل القيم وفق الرمز المعروف باسم EBCDIC* أي أن كل بايت BYTE سوف تحتوي على رمز (حرف أو رقم) واحد. وأحياناً يطلق على طريقة التخزين هذه اسم External or Zoned Decimal .

ويمكن أن تكون عبارة DISPLAY صريحة أو ضمنية أي أن عدم ذكرها مع كلمة USAGE يعني أن الحقل المعرف في قسم البيانات مخزن على أساس رمز واحد في كل بايت.

(*) يقصد بها الكلمات التالية :

Extended Binary Coded Decimal Intercharge Code

وتجدر الملاحظة هنا الى أن طريقة التخزين هذه أساسية عند قراءة البيانات الى البرنامج من بطاقات مثقبة أو عند اخراج البيانات الى الطابعة أو الشاشة الضوئية لأنه في كلتا الحالتين يحتاج كل رمز الى مكان خاص به على وسيلة الادخال (البطاقة) أو وسيلة الاخراج (الصفحة أو الشاشة).

مثال :

```
ϕ5    LIST - FLD     PIC    X (6) USAGE IS DISPLAY.
```

أو

```
ϕ5    E - NUMBER     PIC    9 (5) USAGE DISPLAY VALUE 1234.
```

وهو يعادل :

```
ϕ5    E - NUMBER     PIC    9 (5) VALUE 12345.
```

ففي التوصيفات أعلاه كل رقم يأخذ من الذاكرة داخل الحاسب (بايتا) واحدة.

٥ ـ ٥ ـ ٢ صيغة COMP - 3

من أجل التوفير في السعة التخزينية للأرقام وعوضا عن تخزين كل رقم في بايت واحدة كما في الشكل DISPLAY فانه يمكن استخدام شكل أو صيغة COMP - 3 لتوصيف الحقل الرقمي على أساس أن كل بايت (أو وحدة تخزينية) يمكن أن تتسع لرقمين(*) وهذه الطريقة من التخزين تعرف باسم Packing أو أحياناً Packed decimal. فمثلاً الحقل الموصف في الفقرة السابقة E - NUMBER يحتاج الى ٥ مخازن (بايت) لحفظه في الحاسب بينما عند استخدام شكل COMP - 3 ينخفض العدد الى ثلاثة فقط وتظهر أهمية ذلك عند وجود أماكن تخزين داخلية في البرنامج أو على القرص الممغنط رقمية كبيرة.

(*) من حيث المبدأ فان الأرقام من صفر الى ٩ يمكن تمثيلها بالنظام الثنائي Binary في أربعة مواضع (بت bit) فقط. وبما أن البايت تتألف عادة من ثمانية bit فإنه يمكن قسمها الى منطقتين Zones وكل منطقة تخصص لرقم.

مثال :

```
ϕ5   F - NUMBER   PIC 99   USAGE COMP - 3 VALUE 12.
ϕ5   QNTY   PIC 999   COMP - 3 VALUE 329.
```

فالحقلان E - NUMBER و QNTY يمكن تمثيلهما على الشكل التالي :

Byte		Byte		Byte	
1	2	ϕ	3	2	9
F - NUMBER		QNTY			

وفي الشكل COMP - 3 يجب التصريح عنه إذا أراد المبرمج استخدامه وعدم ذكر هذا الشكل يعني أن الشكل DISPLAY هو المطلوب.

واستخدام COMP - 3 يعتبر مناسباً لاجراء العمليات الحسابية على الحقول الرقمية إلا أنه يجب على المبرمج أن يكون حريصاً على توصيف الحقول المعالجة من نفس الشكل أي أن لا يقوم بعملية حسابية على حقلين احدهما موصوف بـ COMP - 3 والآخر بـ DISPLAY. كما يجب الانتباه الى أن نتائج المعالجة الحسابية للشكل COMP - 3 سوف تكون أيضاً على نفس طريقة التخزين.

وتجدر الاشارة هنا الى أن الشكل COMP - 3 لا يصلح لتوصيف الحقول التي سوف تستخدم لاظهار النتائج على الشاشة أو الطابعة أو لإدخال البيانات من البطاقات المثقبة. ويجب على المبرمج استعمال حقول وسيطة موصوفة بشكل DISPLAY لهذا الغرض.

وعندئذ تقوم تعليمة MOVE آلياً بتوزيع الحقل المخزن على اساس COMP - 3 الى شكل DISPLAY أو العكس تضغط حجم الحقل الموصوف في DISPLAY الى شكل COMP - 3. أن أن تعليمة MOVE تقوم بعملية Pack و Unpack للبيانات المخزنة.

مثال :

```
77    MARK      PIC     9 (5)  COMP - 3.
77    E - MARK      PIC     9 (5).
MOVE MARK TO E - MARK
DISPLAY E - MARK UPON PRINTER.
```

٥ ـ ٥ ـ ٣ صيغة COMP

أما عبارة USAGE IS COMP فإنها تعني أن وصف الحقل الرقمي ممثلا على طريقة النظام الثنائي Binary . ومن مميزات استخدام هذا الشكل في تمثيل الحقل الرقمي في الحاسب امكانية التحكم في تعديل أي «بت» bit في «البايت» Byte الواحدة، وكذلك السرعة في إنجاز العمليات الحسابية على الحقول الرقمية إضافة الى سعة أكثر في التخزين (المكان المخصص للحقل الرقمي).

ويجب ملاحظة أن عملية تحويل (أو نقل) الحقل الموصوف بـ COMP الى شكل DISPLAY غير ضرورية وإنما يمكن اعادة توصيفه بوساطة عبارة REDEFINES واستخدامه في شكله الآخر بعبارة DISPLAY في عمليات الإدخال والاخراج.

مثال :

```
05    BNRY - F       PIC 99 USAGE COMP VALUE 200.
77    BNRY - F2       PIC 999 USAGE COMP VALUE 32000.
```

ففي الحقل الأول BNRY - F فان القيمة 200 المؤلفة من ثلاثة أرقام يمكن وضعها في مخزنين (٢ بايت) لأنه حسب النظام الثنائي يحتويان على ١٦ بت (أو موضع ثنائي).

٥ ـ ٦ عبارة إعادة التسمية RENAMES

إن عبارة إعادة التسمية RENAMES تسمح باعادة تجميع عدة حقول جزئية باسم مختلف عن الاسم الأول الذي عرفت به مسبقاً . وتكتب هذه العبارة في قسم البيانات . وتأخذ هذه العبارة الشكل العام التالي :

```
66    identifier - 1,    RENAMES    identifier - 2

      [ { THROUGH }   identifier - 3 ]
      [ { THRU    }                  ]
```

وهذه العبارة تلي مباشرة الحقول المراد إعادة تسميتها . كما يجب أن تكون الحقول الأصلية والمسماة ثانية في نفس السجل .

وحسب قواعد لغة كوبول* لا يمكن توصيف نفس عبارة اعادة التسمية برقم مستوى 66 ولا يمكن اعادة تسمية المستوى رقم 77 أو 88 أو ϕ1 . كما يجب أيضا أن لا يكون identifier - 2 و identifier - 3 موصوفين بعبارة OCCURS ، أو أي حقل منهما موصوف بهذه العبارة .

أما كلمة THRU وكلمة THROUGH فهما متعادلتان وتعطيان نفس المعنى .

وعند استخدام كلمة THRU يجب أن يكون identifier - 2 يسبق منطقياً الحقل identifier - 3 .

مثال :

```
ϕ1    DATA - RECORD.
   ϕ5    D - FLD - 1    PIC 99.
   ϕ5    D - FLD - 2    PIC X (4).
```

(*) Model 990 Computer COBOL Reference Manual Taxas Instruments, 1980 PP. 5 - 31, 5 - 32

```
ϕ5    D - FLD - 3.
   ϕ7    DFLD - 3A     PIC X.
   ϕ7    DFLD - 3B     PIC X (1ϕ).
ϕ5    D - FLD - 4    PIC X.
66    D - FLD - X    RENAMES D - FLD - 1 THRU DFLD - 3A.
66    D - FLD - Y    RENAMES D - FLD - 2 THRU DFLD - 3B.
```

```
ϕ1    W - AREA.
   ϕ2    W - A       PIC X.
   ϕ2    W - B       PIC 9 (6).
66    CH - CODE RENAMES W - A.
```

٥ - ٧ تعليمة النسخ COPY

تسمح تعليمة النسخ COPY للمبرمج أن يضع في برنامجه أية أجزاء أو اسطر من برامج أخرى في كل من قسم المحيطات ENVIRONMENT أو قسم البيانات DATA أو في قسم الاجراءات PROCEDURE . أو بمعنى آخر تسمح تعليمة COPY بنقل بعض محتويات المكتبة Library (التي تحتوي عادة على برامج أو روتينات) المخزنة على القرص الممغنط .

وأماكن استخدام هذه التعليمة في البرنامج تظهر على النحو التالي :

1) ENVIRONMENT DIVISION

in (CONFIGURATION SECTION) :

SOURCE - COMPUTER COPY (Library - name)

OBJECT - COMPUTER COPY (Library - name)

SPECIAL - NAMES COPY (Library - name)

in (INPUT - OUTPUT SECTION) :

FILE - CONTROL COPY (Library - name)

I - O - CONTROL COPY (Library - name)

2) DATA DIVISION

in (FILE SECTION) :

FD file - name COPY (Library - name)

in a file Description :

01 data - name COPY (Library - name)

(identifier)

PROCEDURE DIVISION

paragraph - name COPY (Library - name)

أمثلة :

```
01    IN - REC COPY PAYROLL.
FD    FILE - 1 COPY INFILE.
PA - 1.    COPY LASTRTN.
```

والأسماء PAYROLL و INFILE و LASTRTN هي عناوين للمكتبة (اسم الملفات) التي تحتوي البيانات المراد نسخها أو نقل صورة عنها الى جانب IN - REC أو FILE - 1 أو PA - 1 .

```
ϕ1    PAYROLL COPY PYLIB.
```

حيث PYLIB :

```
ϕ1    PYLIB.
    ϕ5    F1        PIC X.
```

```
ϕ5    F2         PIC 99.
ϕ5    F3         PIC X (1ϕ).
```

٥ ـ ٨ تجزئة قسم الاجراءات SEGMENTATION

إن تقنية تجزئة قسم الاجراءات في لغة كوبول تسمح بتقسيم البرنامج (Physically) الى أجزاء وكل جزء يدعى Segment وهذا الجزء يشمل مجموعة من التعليمات المترابطة التي تقوم بإنجاز غرض معين أو وظيفة خاصة في البرنامج Particular function.

ومن أهم فوائد التجزئة هذه إعطاء الامكانية للمبرمج لتصميم البرنامج على أجزاء يستطيع التحكم بموجبها في المساحة المتوفرة في الذاكرة لتنفيذ البرامج الطويلة وإبقاء الأجزاء غير اللازمة أثناء عملية تنفيذ البرنامج مخزنة على القرص الممغنط (Overlays). وبالطبع نتيجة رخص قيمة الذاكرات في الوقت الحالي وامكانية استخدام تقنية الذاكرة المنطقية Virtual Storage فقد تقلصت أهمية هذه التجزئة وقل اعتماد المبرمج عليها بالنسبة للسنوات السابقة.

الأجزاء وترقيمها Segments

عند استخدام التجزئة في قسم الاجراءات من قبل المبرمج يمكن وضع مجموعات التعليمات في أجزاء وكل جزء يدعى Segment وكل جزء يتألف من فصول Sections وكل فصل يتألف من فقرات Paragraphs. وللدلالة على تسلسل الأجزاء فإنها تعطى أرقاماً تبدأ من ϕϕ الى الرقم 99(*).

(*) TI990 يستخدم الترقيم من ϕϕ الى 127. ص. 3 - 6

وتأخذ اسم فقرة التقسيم أو التجزئة الشكل العام التالي :

Section - name SECTION [Segment - number].

حيث أن رقم الجزء Segment - number يمكن أن يأخذ الأرقام من ϕϕ الى 99 . ويمكن استخدام خانة واحدة للأرقام التي تكون أقل من 1ϕ . وإذا لم يذكر رقم الجزء بجانب كلمة الفصل SECTION فإنه يفترض أن يكون ϕ .

وهذه الأرقام تأخذ دلالة معينة عند استخدامها فهي تقسم الأجزاء Segment الى نوعين :

أ ـ الأجزاء الثابتة Fixed or Permanent

وهي تلك الأجزاء التي تأخذ الأرقام من ϕϕ الى 49 وتسمى بالأجزاء الثابتة لأنها تبقى موجودة في الذاكرة طيلة مدة تنفيذ البرنامج . ولا يمكن وضع أي منها على أساس Overlay . كما يجب أن تكون متتابعة في قسم الاجراءات وليس من الضروري تسلسل أرقامها .

ب ـ الأجزاء المستقلة (غير الثابتة) Independent

وهي الأجزاء التي تأخذ الأرقام من 5ϕ الى 99* سميت بالأجزاء المستقلة لأنها تعتبر Overlays (أي متروكة جانباً على القرص الممغنط) أثناء تنفيذ البرنامج وتدخل الى الذاكرة الرئيسية عند الحاجة لها بوساطة تعليمة الاستدعاء الخاصة بها مثل PERFORM أو CALL .

وعند استخدام المبرمج لهذه التسهيلات في التجزئة عليه أن يُراعي عدم تشابك الأجزاء الثابتة والأجزاء المستقلة بتعليمة GO TO أو PERFORM وأن يكون تصميمه

(*) الحاسب TI990 يصل الى 127 ص. 3 - 6

للجزء المستقل على أنه جزء متكامل من الفقرات تستدعى بشكل كامل بتعليمة واحدة من قبل الجزء الثابت. كما يجب عليه أن يلحظ الاستخدام المنطقي للتعليمات التي تقوم بنقل التسلسل في البرنامج وبين الأجزاء مثل :

GO TO

PERFORM

ALTER (تعليمة لتغيير مسار GO TO)

كما ينصح قبل استخدام هذه الخاصية في التجزئة الرجوع الى مراجع الحاسب المتوفرة عن لغة كوبول المزودة من قبل الشركة الصانعة بسبب بعض الاختلافات للاستفادة منها.

وأخيراً، وعند استخدام تجزئة قسم الاجراءات الى فصول SECTIONS وفي حالة حاجة المبرمج الاشارة الى فقرة تحمل نفس الاسم موجودة في فصلين يمكن استخدام هذا الشكل العام للدلالة عليها :

$$\text{Paragraph - name} \left[\left\{ \begin{matrix} \underline{\text{OF}} \\ \underline{\text{IN}} \end{matrix} \right\} \text{Section - name} \right]$$

وإذا كانت الفقرات لا تحمل نفس الاسماء في قسم الاجراءات فلا حاجة لهذه العبارة عند الرجوع الى الفقرة المطلوبة.

٥ ـ ٩ البرامج الفرعية والبرامج الرئيسية

تستخدم تعليمة PERFORM لتقسيم البرنامج الواحد الى أجزاء أو فقرات رئيسية وأخرى فرعية لأنها تمكن المبرمج من استدعاء أي جزء من فقرات الاجراءات بما يسهل عملية تحديد الجزء الرئيسي في البرنامج والأجزاء (الروتينات) الفرعية التي يمكن

الاستفادة منها في أكثر من موضع في البرنامج الواحد. وهذا التكوين في البرنامج يطلق عليه اسم Structured Program أي البرنامج المبني على هياكل (هيكل أساسي وأجزاء فرعية).

أما اذا احتاج المبرمج الاستفادة من برنامج كامل آخر في البرنامج الذي يكتبه، فان ذلك يتطلب استدعاء البرنامج الآخر الى برنامجه الأصلي (الرئيسي) وهذا يتوفر في لغة كوبول باستخدام تعليمة CALL (الاستدعاء) والتي تقوم بنقل تسلسل تنفيذ البرنامج الأصلي الى البرنامج الفرعي المُستدعى CALLED ثم متابعة تنفيذ البرنامج الرئيسي بعد تعليمة CALL مباشرة.

وتأخذ تعليمة CALL الشكل العام التالي :

```
       { identifier - 1 }  [ USING  data - name - 1 [, data - name - 2] ... ]
CALL   {                }
       { literal - 1    }  [, ON OVERFLOW imperative - Statement]
```

حيث أن literal - 1 يمثل اسم البرنامج الفرعي المبين في جملة PROGRAM - ID أما استخدام مكان الاسم Identifier - 1 فان ذلك يعين اعطاء اسم متحرك Dynamic أثناء التنفيذ*.

أما عبارة USING الاختيارية فهي تحدد المناطق المشتركة بين البرنامج الرئيسي والبرنامج الفرعي المستدعى وهي اسماء الحقول أو السجلات التي تعطي أو تأخذ البيانات من والى البرنامج الرئيسي والفرعي.

وعند وجود هذه العبارة مع تعليمة CALL يتطلب توصيف فصل خاص للحقول المشتركة مع البرنامج الرئيسي في البرنامج الفرعي و يدعى LINKAGE SECTION**

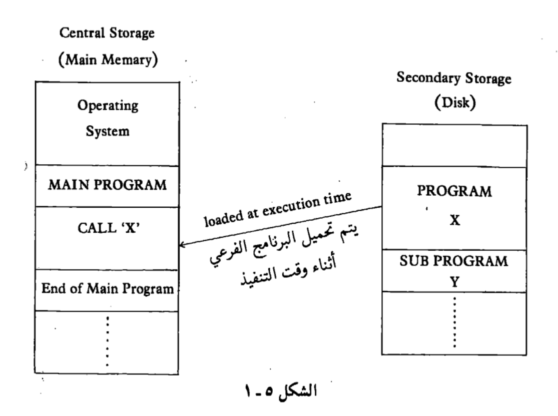
(*) ADVANCED COBOL PP. 270

(**) فصل الربط

و يـأتـي في قـسـم تـوصـيـف الـبيانات (في البرنامج الفرعي) بعد فصل توصيف منطقة العمل الداخلي WORKING-STORAGE .

كـمـا يجـب ذكـر اسـمـاء الحـقـول المـشـتـركة مع عبارة USING الى جانب جملة PROCEDURE DIVISION في الـبـرنـامـج الـفـرعي أيضاً . ولا يشترط أن تكون هذه الأسـمـاء واحـدة في الـتـسمية بل يشترط أن تكون بنفس الترتيب ونفس الحجم والنوع بين البرنامج الرئيسي والفرعي.

أمـا عـبـارة ON OVERFLOW الاخـتـيـاريـة أيضاً فتعني الاشارة الى نقل تسلسل تـنفيذ البرنامج الرئيسي الى الفقرة الواردة فيها عند حصول «فيضان» أو زيادة في حجم البرنامج الفرعي المستدعى Called عن حجم الذاكرة المخصص للتشغيل .



الشكل ٥ - ١

ويتم تحميل Loading البرنامج الفرعي أثناء وقت تنفيذ البرنامج الرئيسي من الذاكرة الثانوية أو الأقراص الممغنطة* الى الذاكرة الرئيسية أو الذاكرة المركزية** فاذا كان المكان المخصص في الذاكرة الرئيسية غير كاف فان تنفيذ البرنامج الرئيسي ينتقل إلى التسلسل المبين بعد جملة ON OVERLOW والشكل (٥ - ١) السابق يوضح عملية استدعاء البرنامج الفرعي أثناء تنفيذ البرنامج الرئيسي.

ولأجل عودة تنفيذ البرنامج الرئيسي في تسلسله المنطقي الأصلي تستخدم تعليمة EXIT PROGRAM في النهاية المنطقية للبرنامج الفرعي عوضاً من تعليمة STOP RUN. وهذه التعليمة تعادل كلمة RETURN في لغات البرمجة الأخرى.

وبعد تنفيذ البرنامج الفرعي بعد استدعائه من قبل البرنامج الرئيسي فقد إشتملت لغة كوبول بعد تطويرها ١٩٧٤ على تعليمة تمكن المبرمج من تخليص الذاكرة الرئيسية من البرنامج الفرعي المنفذ لإعطاء (لاتاحة) المجال لبرنامج فرعي آخر من احتلال مكانه المخصص في الذاكرة الرئيسية والتي غالباً ما تكون محدودة الحجم بالنسبة للذواكر الثانوية. وهذه التعليمة تدعى تعليمة الإلغاء CANCEL وتأخذ الشكل العام التالي:

$$\underline{\text{CANCEL}} \left\{ \begin{array}{l} \text{identifier - 1} \\ \\ \text{literal - 1} \end{array} \right\} \left[\left\{ \begin{array}{l} \text{identifier - 2} \\ \\ \text{literal - 2} \end{array} \right\} \right] \ldots$$

ويقصد بالمتغير identifier أو الثابت literal اسم البرنامج الفرعي المراد الغاؤه من الذاكرة الرئيسية بعد تنفيذه.

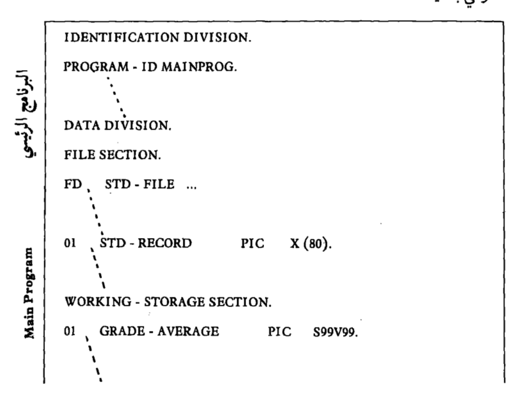
(*) Secondary Storage or Disk

(**) Main Memery or Central Storage.

وتأتي تعليمة الإلغاء بعد تعليمة الاستدعاء في البرنامج الرئيسي ، وقد لا تكون ضرورية في بعض أنواع الحاسبات حيث يقوم نظام التشغيل بعملية الإلغاء هذه بدلا من المبرمج . و ينصح الرجوع الى مراجع الشركة الصانعة عند استخدام طريقة البرمجة المجزئة Modular Programing .

و يوجد في الصفحتين التاليتين مثال عن كيفية استخدام تعليمة الاستدعاء في البرنامج الرئيسي لبرنامج فرعي آخر يدعى SUBPROG .

و يلاحظ من المثال أن ترتيب أسماء الحقول المشتركة بعد عبارة USING واحد في البرنامج وإن اختلفت الأسماء . كذلك فان توصيفهما واحد وكذلك الحجم واحد . كما يلاحظ أيضاً أن البرنامج الرئيسي ينتهي منطقياً بتعليمة STOP RUN والبرنامج الفرعي بتعليمة EXIT PROGRAM .


البرنامج الرئيسي

Main Program

```
IDENTIFICATION DIVISION.
PROGRAM - ID MAINPROG.
       .
       .
       .
DATA DIVISION.
FILE SECTION.
FD    STD - FILE   ...
   .
   .
   .
01    STD - RECORD            PIC      X (80).
   .
   .
   .
WORKING - STORAGE SECTION.
01    GRADE - AVERAGE            PIC      S99V99.
   .
   .
   .
```

```
PROCEDURE DIVISION.
      .
      .
   CALL 'SUBPROG' USING STD - RECORD, GRADE - AVERAGE.
      .
      .
END - OF - MAINP.
      .
      .
   STOP RUN.
```

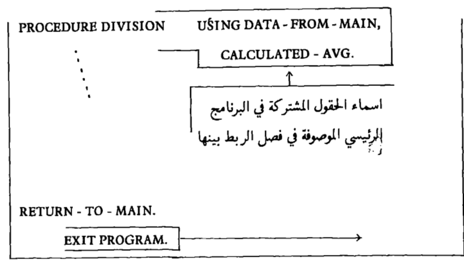
العودة الى ما بعد CALL بعد انتهاء تنفيذ البرنامج الفرعي

استدعاء البرنامج الفرعي المسمى SUBPROG مع استخدام الحقول المشتركة والمسماة في البرنامج الرئيسي STD - RECORD, GRADE - AVERAGE.

البرنامج الفرعي Sub - Program / Called Program

```
IDENTIFICATION DIVISION.
PROGRAM - ID SUBPROG.
      .
      .
DATA DIVISION.
WORKING - STORAGE SECTION.
      .
      .
LINKAGE SECTION
77    CALCULATED - AVG PIC S99V99.
01    DATA - FROM - MAIN      PIC X (80).
      .
      .
```

فصل توصيف الحقول المشتركة في البرنامج الرئيسي



وفي ختام هذا الفصل لا بد من الإشارة الى أن عملية وجود برنامج رئيسي و برامج فرعية أخرى مستدعاة من قبله تتطلب عملية ترجمة كل برنامج على حدة و بشكل مستقل ثم تحتاج الى عملية ربط منطقي بينهما لأجل تعريف الحاسب بأن البرنامج الرئيسي أثناء تنفيذه يحتاج الى برامج (أو برنامج) فرعية أخرى لذلك لا بد من إجراء عملية بعد الترجمة COMPILATION وتسمى بعملية الربط المنطقي Linkedit وهذه العملية تتم بعد خلو البرامج الفرعية والرئيسية من الأخطاء وتصبح جاهزة للاختبار أو التشغيل*.

مثال عملي

في الصفحات التالية نموذج عن برنامج رئيسي و برنامجان فرعيان تم اعدادهما وتنفيذهما على الحاسب IBM لأجل ايضاح كيفية ربط البرامج مع بعضها حيث أن أحد هذين البرنامجين الفرعيين يقوم بعملية الضرب (التربيع) والآخر يقوم بعملية الجمع على نفس القيم وثم تظهر النتيجة بوساطة تعليمات البرنامج الرئيسي.

(*) هناك بعض أنواع من الحاسبات لا تحتاج الى عملية الربط هذه وانما تنفذ آلياً من قبل نظام التشغيل مثل IMOS NCR لذلك ينصح بالرجوع الى مراجع الشركة الصانعة.

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID.     MAINP.
 AUTHOR. ĂJND-KHAYAT.
**       THIS PROGRAM IS TO ILLUSTRATE  **
**          COBOL  CALL  STATEMENT.     **
**           IT IS  MAIN PROGRAM        **
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER. IBM-3033.
 OBJECT-COMPUTER. IBM-3033.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 77    FUNC         PIC X.
 77    WAIT1        PIC X.
 01   AREA-S.
    05  MA1          PIC 99999.
    05  MB1          PIC 9(11).
*
 PROCEDURE DIVISION.
 ST.
      DISPLAY  'ENTER  5 DIGIT-NUMBER '.
      ACCEPT  MA1
      IF MA1 NOT NUMERIC
            MOVE  ZEROES TO MA1.
      DISPLAY
      'SELECT (M)  FOR SQUARE  '
          'OR (A) FOR ADD '
          'OR (9) FOR END '.
      ACCEPT  FUNC.
      IF  FUNC = 'M'
        CALL 'SPROG' USING  AREA-S
        DISPLAY 'THE SQUARE VALUE IS  '
        DISPLAY  MB1
      ELSE
      IF  FUNC = 'A'
        CALL 'SPROG2' USING  AREA-S
        DISPLAY 'THE TOTAL VALUE IS  '
        DISPLAY  MB1
      ELSE
      IF  FUNC = '9'   STOP RUN.
      GO TO ST.
```

البرنامج الرئيسي

Main Program

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID.    SPROG.
 AUTHOR. AJND-KHAYAT.
**       THIS PROGRAM IS TO ILLUSTRATE  **
**          COBOL  CALL  STATEMENT.     **
**       IT IS SUBPROGRAM FOR SQUARE    **
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER. IBM-3033.
 OBJECT-COMPUTER. IBM-3033.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 01    AREA-S.
     05  A1           PIC 99999.
     05  B1           PIC 9(11).
 LINKAGE SECTION.
 01    SAREA.
     05  SA1           PIC 99999.
     05  SB1           PIC 9(11).
*
 PROCEDURE DIVISION  USING  SAREA.
 ST.
      MULTIPLY  SA1  BY  SA1  GIVING SB1.
      DISPLAY  'SUBPROGRAM  SPROG EXECUTED'.
 EXTP.
        EXIT PROGRAM.
```

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID.    SPROG2.
 AUTHOR. AJND-KHAYAT.
**       THIS PROGRAM IS TO ILLUSTRATE   **
**          COBOL  CALL  STATEMENT.      **
**       IT IS SUBPROGRAM FOR ADITION    **
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER. IBM-3033.
 OBJECT-COMPUTER. IBM-3033.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 01   AREA-S.
    05  A1          PIC 99999.
    05  B1          PIC 9(11).
 LINKAGE SECTION.
 01   SAREA.
    05  SA1           PIC 99999.
    05  SB1           PIC 9(11).
*
 PROCEDURE DIVISION USING SAREA.
 ST.
      ADD        SA1,      SA1  GIVING SB1.
      DISPLAY  'SUBPROGRAM  SPROG2 EXECUTED'.
 EXTP.
        EXIT PROGRAM.
*
```

٥ ـ ١٠ اسئلة وتمارين

سؤال ١ ـ بين الخطأ والصحيح من العبارات التالية بالإشارة (✓) أو (x) على التوالي :

- لا يمكن استخدام عبارة REDEFINES مع أي رقم مستوى Level Number
- يمكن أن تأتي عبارة REDEFINES مع رقم المستوى 77
- ترد عبارة RENAMES مع أي رقم مستوى Level Number .
- تحتوي الفقرة Paragraph في قسم الاجراءات على فصل واحد أو أكثر و يسمى SECTION .
- تفيد عملية تجزئة قسم الاجراءات Segmentation في الاستغلال الأمثل للذاكرة المحدودة الحجم في تشغيل البرامج الكبيرة .
- يمكن أن ترد تعليمة CALL في البرنامج الفرعي Subprogram لاستدعاء برنامج فرعي آخر.
- يذكر فصل الربط LINKAGE SECTION في البرنامج الرئيسي Main .
- يمكن استدعاء أكثر من برنامج فرعي بوساطة عدة تعليمات CALL تذكر في البرنامج الرئيسي .
- يسمح بذكر عبارة COPY في قسم البيانات فقط .

سؤال ٢ ـ يبين الجدول التالي توصيف وقيمة الحقل المراد نقله بوساطة تعليمة MOVE الى الحقل المسمى Receiving field . والمطلوب توضيح شكل القيمة الناتجة بعد

عملية النقل :

Sending Field		Receiving Field	
PICTURE	VALUE	PICTURE	EDITED RESULT
S9(4)V99	-456ϕϕ	$$$$$.99CR	
S9(4)V99	456ϕϕ	$$$$$.99DB	
S9(4)	4567	$,$$$.ϕϕ	
S9(6)	121ϕ82	99B99B99	
S9(6)	121ϕ82	99/99/99	
S9(6)	123456	----,---	
9(6)V99	12345ϕ	$ZZZ,ZZZ.99	
9(6)V99	12345ϕ	$***,***99.	
V99	12	VPPP99	

سؤال ٣ ـ في الشكل التالي المأخوذ من برنامج كوبول يوجد توصيف لحقول تم إعادة توصيف لبعضها والمطلوب معرفة :

١ ـ حجم السجل أو الحقل A - AREA بالبايت

٢ ـ ما هي الحقول التي تأخذ نفس المساحة التي يمثلها الحقل P - FLD .

```
ϕ1 A - AREA.
   ϕ2   B - AREA.
      ϕ5   C - FLD        PIC xx.
      ϕ5   D - FLD        PIC xxx.
   ϕ2   E - AREA REDEFINES B - AREA     PIC 9 (5).
```

```
ϕ2   F - AREA.
     ϕ5   G - FLD          PIC 9 (5).
     ϕ5   H - FLD.
          ϕ7   I - FLD          PIC xx.
          ϕ7   J - FLD          PIC 99.
     ϕ5   K - FLD REDEFINES H - FLD.
          ϕ7   L - FLD          PIC x.
          ϕ7   M - FLD          PIC    9 (6) COMP - 3.
ϕ2   N - AREA REDEFINES F - AREA.
     ϕ5   Q - FLD          PIC X (4).
     ϕ5   P - FLD          PIC X (5).
```

تمرين ١

أكتب برنامجاً بلغة كوبول يقرأ ملف المشتركين SUBFILE و يعمل تحققاً للحقول Validation وفق القواعد التالية :

- رقم المشترك D - No يجب أن يكون رقمياً واكبر من الصفر.
- اسم المشترك D - NAME يجب ألا يكون رقمياً ، أو فارغاً .
- عنوان المشترك D - ADRESS يجب ألا يكون فراغاً .
- تاريخ الاشتراك D - DATE :

السنة D - YEAR يجب أن تكون بين ١٤٠٢ و ١٤٠٥

الشهر D - MONTH يجب أن يكون بين ١ - ١٢

اليوم D - DAY يجب أن يكون بين ١ - ٣٠

- رمز الاشتراك D - CODE يجب أن يكون بين ١ - ١٠
- منطقة الاشتراك D - AREA يجب أن تكون محصورة بين ١ - ١٥
- مقدار القراءة D - LREAD يجب أن تكون رقمية
- رقم العداد D - CNTR رقم العداد يجب أن يكون اكبر من الصفر

عند وجود خطأ في أي حقل من الحقول السابقة حرك محتوى السجل الى الطابعة واطبعه وأطبع نجمة * تحت الحقل الخطأ، وكذلك انقل محتوى السجل نفسه الى ملف اخراج آخر اسمه INVFILE موجود على قرص ممغنط. وأعد معالجة سجل جديد.

أما اذا كانت كل الحقول سليمة انقل محتوى السجل الى ملف اخراج اسمه VALFILE على القرص الممغنط ودون طباعته، وأعد معالجة سجل جديد.

ملاحظات الطباعة :

١ - لا تكتب في الصفحة أكثر من ٢٥ سجلاً
٢ - اترك سطراً فارغاً بين كل سطري طباعة
٣ - اكتب العناوين والتاريخ ورقم الصفحة في رأس كل صفحة

توصيف سجل ملف INVFILE :

FIELD NAME	PICTURE
INV - NO	9 (5)
INV - NAME	X (2∅)
INV - ADRESS	X(24)
INV - DATE	9 (8).
INV - CODE	99
INV - AREA	99
INV - CNTR	9 (7)

INV - LREAD	9 (7)
INV - FLR	X (5)

توصيف سجل ملف VALFILE :

FIELD NAME	PICTURE
VAL - NO	9 (5)
VAL - NAME	X (2ϕ)
VAL - ADRESS	X (24)
VAL - DATE	9 (8)
VAL - CODE	99
VAL - AREA	99
VAL - CNTR	9 (7)
VAL - LREAD	9 (7)
FLR	X (5)

توصيف سجل ملف SUBFILE :

FIELD NAME	PICTURE
D - NO	9 (5)
D - NAME	X (2ϕ)
D - ADRESS	X (24)
D - DATE	
D - YEAR	9 (4)
D - MONTH	99
D - DAY	99
D - CODE	99

D - AREA	99
D - CNTR	9 (7)
D - LREAD	9 (7)
D - FLR	X (5)

أنظر صفحة ٦٠٣ لتوصيف شكل المخرجات وصفحة ٦٠٠ للنتائج.

تمرين ٢

ملف رواتب يحتوي على سجلات الموظفين العائدين من الإجازة وموجودة على قرص ممغنط ويحوي المعلومات التالية :

	FIELD NAME	PICTURE
رقم الموظف	EMP - NO	9 (5)
الاسم	EMP - NAME	X (30)
راتبه الأساسي	EMP - SAL	9 (5) V99
بدل نقل شهري	EMP - TRANS	9 (3) V99
بدل تدريب شهري	EMP - ALW	9(5)V99
تاريخ العودة من الإجازة	EMP - DATE	
يوم	EMP - DAY	99
شهر	EMP - MONTH	99
سنة	EMP - YEAR	9(4)
غير مستعمل	FILLER	X (18)

المطلوب كتابة برنامج يجري العمليات التالية :

١ - حساب صافي الراتب بعد احتساب الراتب الأساسي بدل النقل بدل التدريب وذلك عن المدة المتبقية من الشهر (يعتبر الشهر ٣٠ يوماً).

وكذلك حساب التقاعد بواقع ٩٪ عن المبلغ المستحق من الراتب الأساسي.

٢ - طباعة جدول الرواتب.

٣ - استعمل برنامجاً فرعياً SUBPROGRAM لإجراء جميع العمليات الحسابية.

٤ - استخدم تعليمة CALL في البرنامج الأساسي من أجل استدعاء البرنامج الفرعي.

* لا تطبع في الصفحة الواحدة أكثر من ٣٠ اسم، اترك فراغاً بين كل سطري طباعة.

أنظر صفحة ٦٠٤ لتوصيف شكل المخرجات.

الفصل السادس

الجداول في لغة كوبول
TABLES

٦ ـ ١ لمحة عامة

يمكن تعريف الجدول Table* بأنه عبارة عن رموز** مرتبة بطريقة يتكرر فيها نفس الحقل عدة مرات متتالية.

فمثلاً هناك عدة بنود في فاتورة يمكن وصفها على شكل جدول حيث أن البند يتكرر في الفاتورة أكثر من مرة وقد يحتوي هذا البند على رموز السلعة أو وصفها مع السعر والقيمة. كذلك يمكن وضع رموز أسماء المناطق أو المدن مع أسمائها على شكل جدول مثل :

```
ϕ1   RIYADH
ϕ2   JEDDAH
ϕ3   DAMMAM
```

حيث أن البند*** (أو الصف) الواحد من الجدول يقصد به رمز المدينة مع اسمها وذلك لتسهيل عملية تداول ومعالجة هذه البيانات الموجودة في الجدول ببرنامج الكوبول. وسوف يظهر في الصفحات التالية بعد شرح لوصف وكيفية استخدام

(*) Programmers COBOL Guide

(**) يقصد بها Characters أي أحرف وأرقام.

(***) table entry أو table item

الجداول في الكوبول أهمية الجدولة في تسهيل عمل المبرمج واختصار العديد من الخطوات التي قد يحتاجها بدون عملية الجدولة لهذه البيانات وخاصة تلك التي يمكن صياغتها بشكل سهل ومبسط على أشكال جداول.

وتقسم أحيانا الجداول حسب أبعادها فهناك بعد واحد (أي صفوف فقط) وهناك بعدين (صف وعمود) أو ثلاثة أبعاد (صف وعمود وارتفاع كما في أشكال الحجوم).

وقبل الدخول في الأشكال الخاصة بكوبول لا بد من ننوه الى أن الصف أو العمود في الجدول الخاص بكوبول يمكن أن يحتوي على حقل أو أكثر data - name. كما أن هناك طريقتين أو اسلوبين مختلفين في كوبول لمعالجة وتداول بيانات الجدول وهما :

اسلوب :

- استخدام المؤشر Subscripting
- استخدام الدليل Indexing

ولكل من هذين الاسلوبين مزاياه ولكن لا بد للمبرمج من التمييز بينهما حيث أنهما متشابهان أو مترادفان في لغات البرمجة الأخرى.

تعريف الجدول وعبارة التكرار OCCURS

يقصد بكلمة OCCURS* في هذا المجال تكرار حدث معين مرات عديدة بنفس الشكل، وهذا ما يمكن تطبيقه على عملية تكرار نفس السطر أو العمود في الجدول لأنه يأخذ نفس الشكل، لذلك فإن لغة كوبول تستخدم عبارة OCCURS للدلالة على

(*) تعني OCCURS يوجد أو يحدث، يظهر (المورد طبعة ١٩٨٢)

تـعـريـف الجـدول في قـسـم البيانات. وتأخذ هذه العبارة عدة صيغ عامة نذكر أبسطها وهي الصيغة التالية :

(Level - number) (data - name) <u>OCCURS</u> (integer) TIMES.

ورقم المستوى Level - number يمكن أن يكون أي رقم مستوى عدا 01 أو 77 أو 88 أو 66 أي ينحصر بين 02 و 49* ، ويمكن استخدام عبارة OCCURS لتوصيف أي جزء مـن الـسـجـل Record الخاص بالملف ولكن لا يمكن استخدامها لتوصيف سجل أو بند Item مستقل بذاته.

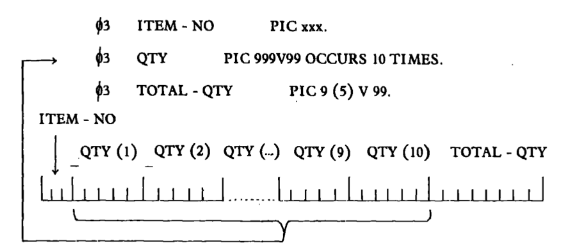
أما عـدد مـرات تـكـرار الـصـف أو الـبـنـد في الجـدول فهو يمثل في العدد الصحيح Integer ويجـب أن يـكـون عدداً موجباً ويختلف الحد الأقصى له باختلاف نوع الحاسب المستخدم فبعضها يصل الى 32767** .

كما أن عبارة OCCURS تصلح لتعريف الحقل أو البند الأ ولي ELEMENTARY أو مجـمـوعـة مـن الحـقـول GROUP. على أن لا تحتوي عبارة OCCURS عبارة VALUE المـسـتـخـدمـة لـتـوصـيـف القيمة الثابتة. وتجدر الاشارة الى أنه لا يمكن إستخدام إعادة تـعـريـف الحـقـل المـوصـوف بـعـبـارة OCCURS بـوسـاطـة عـبـارة إعـادة الـتـعـريـف REDEFINES بـل الـعـكـس جـائـز وممكن. أي اعادة تعريف أي حقل باسم آخر مع عبارة OCCURS.

(*) The Programmer's ANSI COBOL, Ref. Manual Donald A. Sordillo, Prentice - Hall 1978 PP. 166

(**) NCR IMOS صفحة ٦٦

الشكل التالي يبين استخدام OCCURS في التعريف :



تعريف الجدول ذو أكثر من بُعد NESTED OCCURS

إن الجـدول المـوصـوف بـعـبـارة واحدة OCCURS يعتبر جدولا ذا بُعد واحد وظهور عبـارة تالية OCCURS في مستوى Level أدنى وأقل تابع للعبارة الأ ولى يدل على وجود جـدول فـرعـي آخـر يتبع للجدول الأ ول. وهذا ما يسمى بالجدول ذي البعدين (صف وعـمود) أو هناك في الجدول نوعان من التقسيمات : تقسيمات اجمالية مثلاً : أقسام في مـؤسـسـة، ثم تقسيمات فرعية : مثلاً : شُعب متعددة تتبع هذه الأقسام. فمثلاً الشكل الـتـالي بـجدول يحتوي أربعة أقسام للبرامج التدريبية في معهد الادارة العامة وكل قسم يحـتـوي على عـدد مـن الـبرامج يصل في المثال هذا الى خمسة برامج فيمكن توصيف هذا الجدول كما يلي :

القسم (1) برامج عليا					القسم (2) برامج اعدادية					القسم (3) برامج خاصة					القسم (4) برامج تدريبية				
(1)	(2)	(3)	(4)	(5)	(1)	(2)	(3)	(4)	(5)	(1)	(2)	(3)	(4)	(5)	(1)	(2)	(3)	(4)	(5)

```
Ø1    EXAMPLE - TAB - 2.
   Ø5    DIVISIONS    OCCURS 4 TIMES.
      Ø1    PROGRAMS OCCURS 5 TIMES    PIC 99.
```

أما اذا تواجد في توصيف الجدول ثلاثة عبارات لـ OCCURS وكل واحدة منها تابعة للسابقة لها في تسلسل رقم المستوى فعندئذ يكون هذا الجدول ذا ثلاثة أبعاده.
والجدول التالي يعطي صورة لجدول مكون من ثلاثة أبعاد :

FLDX	FLDY (1)		FLDY (2)		FLDZ
	FLDY - A (1)	FLDY - A (2)	FLDY - A (3)	FLDY - A (4)	
	(1) (2)	(1) (2)	(1) (2)	(1) (2)	

FLDY - B

ويمكن توصيفه كما يلي :

```
Ø1    EXAMPLE - TAB - 3.
   Ø5    FLDX         PIC xxx.
   Ø5    FLDY      OCCURS 2 TIMES.
      1Ø    FLDY - A     OCCURS 2 TIMES.
         15    FLDY - B     OCCURS 2 TIMES.
            2Ø    FLDYB1        PIC X.
            2Ø    FLDYB 2       PIC XX.
   Ø5    FLDZ         PIC XXX.
```

(*) لا يستخدم في لغة كوبول أكثر من ثلاثة أبعاد.

و يلاحظ من الشكل السابق أن عدد تكرار حدوث البنود FLDY هو مرتان (1) FLDY و (2) FLDY أي أن FLDY هو جدول ذو بعد واحد أما الحقل أو البند FLDY - A فهو مكرر مرتين ضمن الجدول الأصلي FLDY أن أن هناك أربعة حقول جزئية كما يلي :

FLDY - A (1,1), FLDY - A (1,2) FLDY - A (2,1) FLDY - A (2,2)

والأرقام الموجودة بين قوسين تدل على تسلسل أو ترتيب العنصر في الجدول فالعنصر (أو البند) (1,2) FLDY - A يعني أن العنصر الأول من الجدول الأول أي أول عبارة OCCURS وهو في نفس الوقت العنصر الثاني من الجدول الفرعي أي ثاني عبارة OCCURS أو كما يوصف في عمليات المصفوفات بدلالة السطر والعمود على التوالي.

وعند أخذ عبارة OCCURS الثالثة في الاعتبار يصبح لدينا البنود الجزئية التالية :

FLDY - B (1,1,1) , FLDY - B (1,1,2)

FLDY - B (1,2,1) , FLDY - B (1,2,2)

FLDY - B (2,1,1) , FLDY - B (2,1,2)

FLDY - B (2,2,1) , FLDY - B (2,2,2)

حيث أنه كل بند من هذه البنود يحتوي حقلين جزئيين وهما :

FLDY - B1 (I,J,K)

و FLDY - B2 (I,J,K)

وللإشارة الى أي عنصر أو بند في الجدول سواء كان ذا بُعد واحد أو أكثر تستخدم الأقواس والرقم الخاص بترتيب هذا البند أو العنصر بينهما و يطلق على هذا الرقم اسم المؤشر أو الدليل* Subscript أو Index وهناك اختلاف بين الأثنين في لغة كوبول كما سوف يلاحظ في الصفحات التالية.

(*) قد يستعمل أحيانا باسم الرمز أو الدليل السفلي لأنه يذكر في رموز الرياضيات في أسفل وجانب المتغير مثل : س١ ، س٢ ، س٣ ...

٦ ـ ٤ تداول الجداول ACCESSING TABLES

تتم عملية تداول عناصر الجدول المعرف بعبارة أو عبارات OCCURS بوساطة تحديد ترتيب وموضع هذا العنصر في الجدول فإذا كان الجدول ذا بُعد واحد يحدد رقم السطر أو البند الذي يحتوي العنصر وهذا الرقم نسبي يبدأ من الرقم ١ والى نهاية حجم الجدول. وإذا كان الجدول ذا بُعدين يحدد رقم السطر ثم رقم العمود الذي يدل على العنصر أو البند في الجدول وكذلك هذه الأرقام نسبية لأنها تبدأ من الرقم ١ الى نهاية حجم الجدول. وكذلك إذا كا ن الجدول ذا ثلا ثة أبعاد تحدد المؤشرات الثلا ثة التي تبين موضع العنصر في الجدول، لأن جميع عناصر الجدول المكررة في عبارة OCCURS تحمل نفس الاسم ولأجل تمييزها عن بعضها يشار اليها بموضعها النسبي في الجدول Relative Position . وتوجد طريقتان في لغة كوبول من أجل تداول عناصر الجدول وتحديد موضع هذه العناصر وهما :

- استخدام المؤشر Subscripting
- استخدام الدليل Indexing

٦ ـ ٤ ـ ١ استخدام المؤشر SUBSCRIPTING

بعد توصيف الجدول Table ومن أجل الإشارة الى عنصر أو بند في هذا الجدول يستخدم حقل رقمي يدعى المؤشر Subscript عُرفاً و يوصف في قسم البيانات في فصل التوصيف الداخلي في البرنامج كأي حقل آخر مع ملاحظة أن حجمه يجب أن يتناسب مع حجم أو عدد العناصر المكونة للجدول. كما يمكن الدلالة على المؤشر مباشرة باستخدام رقم ثابت صحيح يمثل المكان النسبي للعنصر في الجدول ومن أجل تمييزه عن غيره من الثوابت أو الحقول يوضع بين قوسين بعد ذكر اسم البند المكرر في الجدول

مثل : QTY (1) , QTY (5)

للدلالة على العنصر رقم (1) والعنصر رقم (5) في الجدول الموصوف في الصفحات السابقة.

أو عند استخدام حقل وسيط لتخزين مكان العنصر مثل الاسم SUB1 يكون :

```
WORKING - STORAG SECTION.
77     SUB1        PIC 99.
       \
        \
PROCEDURE DIVISION.
  \
   \
    \
    MOVE  1 TO SUB1
    MOVE  QTY (SUB1) TO ....
    MOVE  5 TO SUB1  ....
    MOVE  QTY (SUB1) TO ....
```

وإذا كان الجدول ذا أكثر من بُعد واحد يكون الشكل التالي للدلالة على العنصر فيه :

FLDY - A (2,1)

أو FLDY - A (S1,C1)

حيث يدل الرقم الأول على رقم السطر والرقم التالي على رقم العمود في الجدول و يأخذ استخدام المؤشر للدلالة على عنصر الجدول الشكل العام التالي :

```
* { data - name - 1     } [ { OF } data - name - 2 ] ....
  { Condition - name    } [ { IN }                 ]
        (Subscript - 1 [, Subscript - 2 [, Subscript - 3]] )
```

(*) لا يسمح نظام TI990 بوجود Condition - name

حيث أن المؤشر يوضع بين قوسين و يوجد فراغ قبل وآخر بعد القوس الثاني للفصل بينه و بين أسماء الحقول المجاورة. وإذا وُجد أكثر من مؤشر (جدول ذو أكثر من بُعد) توضع المؤشرات على الترتيب حسب التسلسل المنطقي لوضع العنصر في الجدول و يفصل بينها فاصلة أو فراغ* .

كما تجب الإشارة الى أن المؤشر لا يمكن أن يكون مقبولاً اذا كان هو بحد ذاته جدولا ويحتاج الى مؤشر آخر له. أي يكون Subscripted Subscript .

وفي حالة كون المؤشر المستخدم عدد غير ثابت أي اسم الحقل فإنه يمكن إجراء العمليات الحسابية عليه وكذلك تعليمات المقارنة IF .

كما أن هناك بعض أنواع من الحاسبات** تسمح بأن يكون هذا المؤشر عدداً جبرياً أي يحتوي على إشارة جبرية إلا أنه يجب أن لا يقل عن قيمة 1 وأن يكون عدداً موجب الإشارة.

٦ ـ ٤ ـ ٢ استخدام الدليل INDEXING

إن استخدام الدليل أو الفهرس Index يشبه الى حد كبير استخدام المؤشر من أجل الدلالة على موضع العنصر في الجدول وتداول مع بعض الاختلافات ويمكن تلخيصها فيما يلي :

- يضاف الى عبارة OCCURS جملة ثانية هي INDEXED BY من أجل تحديد اسم الدليل الخاص بالجدول المعّرف بـ OCCURS و يصبح الشكل العام لهذه العبارة :

(level - number) (datat - name) <u>OCCURS</u> (integer) TIMES

[<u>INDEXED</u> BY index - name - 1 [index - name - 2] ....]

(*) Donald A. Sordillo PP. 317

(**) TI990 صفحة 46 - 5

حيث أن كلمة index - name تدل على اسم يحدد من قبل المبرمج للدلالة على الدليل المستخدم مع عبارة OCCURS وهذا الدليل عبارة عن حقل يوصف آلياً من قبل نظام التشغيل (مترجم كوبول) ولا حاجة لتعريفه من قبل المبرمج في قسم البيانات و يتعامل نظام التشغيل معه بقيمة العدد الثنائي Binary Value من أجل تحديد مواضع العناصر في الجدول.

- تستخدم تعليمة خاصة تدعى SET لزيادة أو تخفيض قيمة الدليل Index - name وكذلك لاعطائه القيمة الأولية (الابتدائية) ولا يمكن استخدام التعليمات الحسابية ... , ADD, SUBTRACT أو تعليمة النقل MOVE لأجل تعديل قيمته.
- يجب استخدام نفس أسماء الأدلة المذكورة مع عبارة OCCURS للدلالة على العنصر في الجدول ولا يجوز استخدام أسماء حقول أو متغيرات أخرى ولذلك يجب تحديد قيمة هذا الدليل بخطوة سابقة لتعليمة تداول العنصر في الجدول.
- يمكن استخدام الجدول الموّصف مع دليل Index - name بتعليمة SEARCH أما وجود المؤشر Subscript فهو لا يسمح باستعمالها.

ويمكن اعادة توصيف الجدول المذكور في الفقرة السابقة باستخدام الدليل على الشكل التالي :

```
ϕ1    EXAMPLE - TAB - X3.
   ϕ5    FLDX    PIC xxx.
   ϕ5    FLDY    OCCURS    2 TIMES
              INDEXED BY      XD1.
      1ϕ    FLDY - A    OCCURS    2 TIMES
              INDEXED BY XD2.
         15    FLDY - B    OCCURS    2 TIMES
              INDEXED BY XD3.
```

```
        20    FLDYB1     PIC X.
        20    FLDYB2     PIC XX.
    05    FLDZ    PIC XXX.
```

حيث أن XD1 , XD2 , XD3 تكون أسماء لأدلة الجدول فإذا أردنا تداول أي بند في الجدول فيجب ذكر هذا الدليل للدلالة على موضع هذا البند مثل :

```
FLDY     (XD1)
FLDY - A (XD1 , XD2)
FLDY - B (XD1, XD2, XD3)
```

ولأجل إعطاء قيمة لهذه الأدلة تستخدم سابقاً لها تعليمة SET وسيأتي ذكرها في الفقرة التالية .

عبارة USAGE IS INDEX

لقد مرّ سابقاً أن عبارة USAGE IS توّصف الحقل في ثلاثة أشكال أو صيغ أساسية وفي استخدام الدليل يضاف اليها الشكل الرابع INDEX و يصبح الشكل العام لهذه العبارة على النحو التالي :

```
           ⎧ DISPLAY ⎫
           ⎪ COMP    ⎪
USAGE IS   ⎨ COMP - 3⎬
           ⎪ INDEX   ⎪
           ⎩         ⎭
```

وعبارة USAGE IS INDEX تعرّف الحقل data - name الموصوف بها على أنه حقل في صورة دليل يستخدم لأجل تخزين قيمة الدليل المذكورة بعد عبارة OCCURS بشكل مؤقت لأن كلمة INDEX بعد USAGE تعطيه (أي الحقل) نفس صورة الدليل

الأصلية أي ثنائي القيمة Binary . ولا يحدد المبرمج حجم الحقل الموصوف كدليل بل يترك ذلك الى نظام التشغيل المستخدم (أو الحاسب المستخدم) لأن حجمه يختلف باختلاف الحاسب*.

مثال :

77 INDEX - STORE USAGE IS INDEX

فالحقل INDEX - STORE يأخذ نفس صيغة الدليل المعرف بعد عبارة INDEXED BY . و يعرف باسم Index data item ويمكن أن يكون موصوفاً ضمن مجموعة من الحقول Group ، أو يوّصف مجموعة أدلة .

و يكون الشكل العام لاستخراج الدليل Index مع عناصر الجدول وفق الشكل التالي :

```
** { data - name - 1     } [ { OF }   data - name - 2 ] ...
   { Condition - name    } [ { IN }                   ]

( { index - name - 1   [ { + }   literal - 2 ] }
  { literal - 1          { - }               }

[ , { index - name - 2   [ { + }   literal - 4 ] } ]
    { literal - 3          { - }               }

[ , { index - name - 3   [ { + }   literal - 6 ] } ] )
    { literal - 5          { - }               }
```

(*) يأخذ أربعة بايت ثنائية القيمة في نظام التشغيل NCR IMOS صفحة ٩٧ .

(**) Donald A. Sordillo P. 100 - 101

لا يوجد Condition - name في نظام TI 990

حيث يدل index - name على اسم الدليل المستخدم مع عبارة OCCURS المصحوبة بجملة INDEXED BY .

الفهرسة المباشرة DIRECT INDEXING

و يقصد بهذه الطريقة تداول عنصر الجدول مع ذكر اسم الدليل الخاص به (أو الأدلة الخاصة به) بين قوسين تماماً مثل استخدام المؤشر في الفقرات السابقة.

فمثلاً : FLDY (XD1)

تشير الى العنصر (أو البند) FLDY الذي موضعه في الجدول XD1 .

فإذا كانت قيمة XD1 تساوي 2 فإن العنصر (2) FLDY سيكون محل التداول في التعليمة المستخدمة و بشكل مباشر.

الفهرسة النسبية RELATIVE INDEXING *

وتعني هذه الطريقة من استخدام الدليل إضافة أو طرح قيمة ثابتة عددية من قيمة الدليل الموجود بين القوسين للدلالة على مكان العنصر (أو البند) في الجدول . ويجب أن تكون عملية الإضافة أو الطرح من الدليل متفقة مع الشكل العام المذكور في الفقرة السابقة (يأتي العدد الثابت بعد الدليل وتفصل بينهما الإشارة الحسابية + أو - وتقع بين فراغين) . فمثلاً :

إذا كان الدليل XD1 يساوي 1

FLDY (XD1 + 2)

يقصد بهذا الاسم العنصر (أو البند) FLDY رقم 3 في الجدول . أما :

FLDY (XD1 - 1)

(*) Donald A. Sordillo P. 101

فيقصد به العنصر أو البند رقم 2 إذا كانت قيمة الدليل XD1 تساوي 3 .

ويجب هنا الانتباه الى عدم الوصول الى القيمة ∅ (صفر) أثناء الطرح .

كما أن وجود أكثر من دليل بين القوسين يجب أن يكون حسب الترتيب المنطقي للجدول .

٦ ـ ٥ تعليمة تحديد قيمة الدليل SET*

تستخدم تعليمة SET لإعطاء قيمة عددية للدليل INDEX الموصوف بعبارة INDEXED BY أو لأجل نقل قيمة هذا الدليل الى حقل آخر مسمى كدليل بعبارة USAGE IS INDEX . وكذلك لتعديل قيمة الدليل أيضاً زيادة أو نقصان من أجل تداول العناصر المكررة في الجدول المطلوب .

وتأخذ تعليمة SET الشكل العام التالي وعلى صيغتين :

```
1)
SET  { index - name - 1 [index - name - 2]  } ... TO
     { identifier - 1 [identifier - 2]      }

          { index - name - 3 }
          { identifier - 3   }
          { literal - 1      }
```

حيث أن index - name - 1 و index - name - 2 و ...

هي أسماء للأدلة المذكورة بعبارة OCCURS مع جملة INDEXED BY أما identifier - 1 و identifier - 2 و ... هي أسماء لحقول موصوفة كدليل بعبارة USAGE IS INDEX تستخدم لأجل حفظ قيمة الدليل أو إعادة القيمة إليه .

(*) من معاني كلمة SET باللغة الانكليزية : يضع ، يحدد ، يعين ، يهيىء ـ المورد ١٩٨٢

أمثلة :

لاعطاء القيمة 1 للدليل : SET XD1 TO 1

لحفظ قيمة الدليل في الحقل A1 : SET A1 TO XD1

حيث : 77 A1 USAGE IS INDEX

SET XD2 TO A1

لإعطاء الدليل XD2 القيمة المحتواة في الحقل A1 .

أما الصيغة الثانية لتعليمة SET فتأخذ الشكل التالي :

2)

```
SET    index - name - 1    [index - name - 2] ....

    { UP BY   }  { identifier - 4 }
    { DOWN BY }  { literal - 2    }
```

حيث أن identifier - 4 أي حقل موصوف في قسم البيانات يحتوي على قيمة عددية تستخدم لزيادة أو نقص قيمة الدليل index - name - 1 أو أكثر من دليل في نفس التعليمة .

و يستفاد من هذا الشكل في تعديل قيمة الدليل زيادة أو نقصاناً من أجل الوصول الى الموضع اللازم للعنصر المطلوب في الجدول .

أمثلة :

لزيادة قيمة الدليل XD3 بالقيمة 1 : SET XD3 UP BY 1.

ولزيادة قيمة الدليل XD2 بالعدد 2 : SET XD2 UP BY 2.

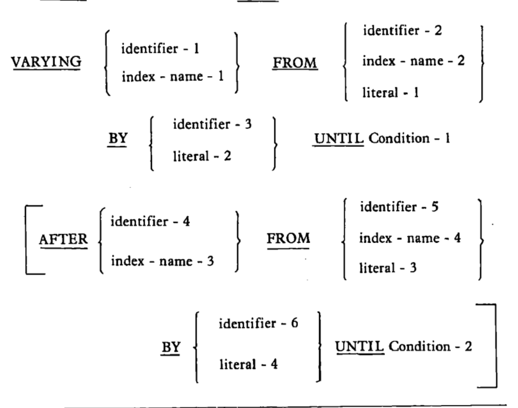
ولانقاص قيمة الدليل XD1 بالقيمة الموجودة في الحقل INCR :

SET XD1 DOWN BY INCR.

تعليمة أنجز المتبدلة PERFORM VARYING*

إن تعليمة PERFORM مع عبارة VARYING من التعليمات المتقدمة في لغة كوبول وهي مناسبة بشكل خاص عند استخدام الجداول في توصيف البيانات، وتصلح سواء كان الجدول موصوفاً مع دليل Indexed أو بدون دليل أي مع مؤشر Subscript . كما يمكن أن تكون ذات فائدة أكبر كلما كبر حجم الجدول ويمكن استخدامها مع البعد الواحد أو البعدين أو ثلاثة أبعاد.

وتأخذ تعليمة «أنجز» PERFORM مع VARYING الشكل العام التالي :



```
PERFORM procedure - name - 1 [THRU procedure - name - 2]

                                             { identifier - 2   }
VARYING { identifier - 1   }   FROM          { index - name - 2 }
        { index - name - 1 }                 { literal - 1      }

        BY { identifier - 3 }   UNTIL Condition - 1
           { literal - 2    }

                                             { identifier - 5   }
[ AFTER { identifier - 4   }   FROM          { index - name - 4 }
        { index - name - 3 }                 { literal - 3      }

        BY { identifier - 6 }   UNTIL Condition - 2 ]
           { literal - 4    }
```

(*) ADVANCED COBOL PP. 401 and

Pragmatic COBOL PP. 172 - 173

```
[ AFTER { identifier - 7    } FROM { identifier - 8   } BY
        { index - name - 5  }      { index - name - 6 }
                                   { literal - 5      }

        { identifier - 9 }   UNTIL Condition - 3 ]
        { literal - 6    }
```

حيث أنه يقصد بالشروط Condition - 1 و Condition - 2 أو Condition - 3 عمليات المقارنة الوارد ذكرها مع تعليمة IF .

ولتوضيح كيفية استخدامها نأخذ المثال التالي :

لدينا جدول يحتوي على ثلاثة معاهد وكل معهد يشمل خمس مدارس وكل مدرسة لها نظام أربع سنوات . وفي كل سنة يوجد عدد من الطلاب . وموصوف هذا الجدول على النحو التالي :

```
ϕ1   ENROLLMENTS.
  ϕ5   COLLEGE  OCCURS 3 TIMES.
    1ϕ   SCHOOL OCCURS 5 TIMES.
      15   YEAR - S OCCURS 4 TIMES.
        2ϕ   STUDENTS     PIC   9 (4).
```

وعند جمع عدد جميع الطلاب في الجدول يمكن استخدام تعليمة PERFORM وفق الشكل التالي :

```
MOVE ZERO TO TOTAL - STUDENTS.
PERFORM SUMMATION VARYING I FROM 1 BY 1 UNTIL I > 3
        AFTER   J   FROM   1 BY 1 UNTIL J > 5
        AFTER   K   FROM   1 BY 1 UNTIL K > 4.
```

حيث أن فقرة SUMMATION تعادل الشكل :

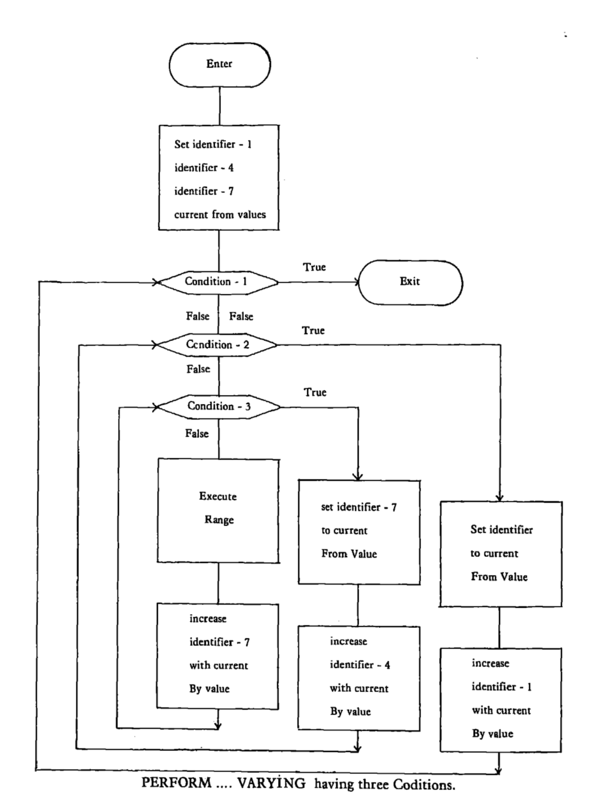
```
SUMMATION.
        ADD STUDENTS (I, J, K)
                TO TOTAL - STUDENTS.
```

وأن I, J, K أسماء للمؤشرات الخاصة بالجدول، كما يمكن أن تكون أسماء للأدلة Indices إذا كان الجدول موصوفاً مع جملة INDEXED BY .

و يلاحظ أن إعطاء القيمة الأولية للمؤشرات أو الأدلة يتم ضمن التعليمة وكذلك المقارنة على نهاية (حجم) الجدول ولكل بُعد من أبعاده.

كما يجب التنويه إلى أن جميع المؤشرات والأدلة يجب أن تكون قيمتها أكبر من الصفر.

والصفحة التالية توضح المخطط الانسيابي لكيفية تنفيذ تعليمة PERFORM مع VARYING بوجود ثلاثة شروط.



PERFORM .... VARYİNG having three Coditions.

٦ ـ ٧ تعليمة البحث في الجدول SEARCH

من أجل تسهيل عملية البحث في الجدول عن عنصر محدد فقد اشتملت لغة كوبول على تعليمة خاصة تسمى SEARCH ويمكن أن تأخذ هذه التعليمة الصيغتين العامتين* :

1)

```
SEARCH identifier - 1  [ VARYING { index - name - 1 } ]
                                 { identifier - 2   }

       [AT END imperative - statement - 1]

       WHEN Condition - 1  { imperative - statement - 2 }
                           { NEXT SENTENCE              }

     [ WHEN Condition - 2  { imperative - statement - 3 } ] ...
     [                     { NEXT SENTENCE              } ]
```

2)

```
SEARCH ALL identifier - 1

       [AT END imperative - statement - 1]

       WHEN Condition - 1  { imperative - statement - 2 }
                           { NEXT SENTENCE              }
```

وفي كلا الصيغتين يشترط أن يكون الجدول محل البحث موصوفاً بعبارة OCCURS والتي تحتوي على جملة INDEXED BY .

(*) Crauer, Crawford صفحة ١٦٤

وتتعامل تعليمة SEARCH مع الجدول على بُعد واحد فقط وإذا كان الجدول له أكثر من بُعد فان البحث يُفترض أن يتم على آخر عبارة OCCURS مذكورة في توصيف الجدول المتعدد الأبعاد بينما تبقى بقية الأبعاد ذات المستويات العليا ثابتة.

وجملة WHEN تعني وجود احتمالين لنتيجة البحث بوساطة التعليمة : إما الحصول على البند (السطر) المطلوب أو عدم الحصول عليه. والشرط الذي يلي كلمة WHEN (يسمى Condition) هو الذي يقرر البند اللازم والمطلوب في البحث. وعند تحقق الشرط ينقل تسلسل التنفيذ الى التعليمة التالية لتعليمة SEARCH في البرنامج ما لم يذكر خلاف ذلك بدلا من جملة NEXT SENTENCE.

أما في حالة البحث عن كامل الجدول وعدم تحقق الشرط (أي عدم الحصول على البند المطلوب) ينتقل التسلسل في التنفيذ الى جملة AT END والتعليمة التالية لها مباشرة.

كما يمكن استخدام جملة NEXT SENTENCE بعد جملة AT END لنقل التسلسل الى التعليمة التالية لـ SEARCH مباشرة إذا رغب المبرمج في ذلك.

وإضافة جملة VARYING الى تعليمة البحث SEARCH تمكن المبرمج من استخدام دليل ثانوي غير الدليل الأولي Prime ليكون عدداً في عملية اجراء البحث. وأحياناً يكون هذا الدليل الثانوي دليلاً في جدول آخر وإذا لم يكن معرفاً بعبارة INDEXED BY فيجب أن يكون موصوفاً لحقل identifier - 2 بعبارة USAGE IS INDEX.

أما الحقل المسمى في الصيغة العامة : identifier - 1 وهو مجال البحث فهو عبارة عن اسم البند Item أو Entry الأساسي في الجدول والذي وصف مع عبارة OCCURS وجملة INDEXED BY. ولا يجوز استخدام البحث على غيره من حقول الجدول وخاصة تلك التي لم يرد وصفها في عبارة OCCURS.

٦ ـ ٧ ـ ١ البحث التتابعي SEQUENTIAL SEARCH

تقوم تعليمة SEARCH بإجراء البحث على التتالي عندما لا تستخدم معها كلمة ALL، وعلى المبرمج أن يحدد نقطة بداية البحث في الجدول بإعطاء القيمة الأولية للدليل الخاصة بالبند محل البحث. وفي عملية البحث تزاد قيمة الدليل بالقيمة (1) آلياً ليتم الانتقال الى البند التالي وهكذا حتى يتحقق شرط الحصول في كلمة WHEN أو الوصول الى نهاية الجدول. وتتم المقارنة على نهاية البحث (قيمة الدليل العظمى) آلياً أيضاً ضمن هذه التعليمة.

والشكل التالي يبين مثالا عن البحث التتابعي :

البحث التتابعي بتعليمة SEARCH

```
                 9.15.45           DEC 25, 1982
001506/
001507  01 TABLE - AREA.
001508     05 QQ - Y OCCURS 10 TIMES INDEXED BY XY.
001509        10 QQ - Z OCCURS 10 TIMES INDEXED BY XZ,
001510                                                  X - ZZ.
001511        15 QQ - Z - 1    PIC X.
001512        15 QQ - Z - 2    PIC S9999.
001513        15 QQ - Z - 3    PIC S9999    COMP.
001514 *         ------------------
001515  01 LOOK - FOR PIC S9 VALUE + 3.
001516  01 MAX - Y        PIC    S9 (3) VALUE    + 10.
001517  01 MAX - Z        PIC    S9 (3) VALUE    + 10.
001518  01 CODE - IS         PIC X.
```

```
001519 *      -----------
001910 *
001920 P - INIT.
001930          PERFORM LOAD - ITEM VARYING XY FROM 1 BY 1
001940              UNTIL XY IS > MAX - Y.
001950 *    ........
001960 *        ........
001970          PERFORM FIND - ITEM VARYING XY FROM 1 BY 1
001980              UNTIL XY IS > MAX - Y.
001990          DISPLAY CODE - IS UPON CONSOLE.
001991          GO TO NXT - SEC.
001992 *
001993  FIND - ITEM SECTION.
001994  F - I - 1.
001995              SET XZ TO 1.
001996 *
001997          SEARCH QQ - Z
001998              AT END MOVE 'N' TO CODE - IS
001999              WHEN QQ - Z - 2 (XY, XZ) IS  = LOOK - FOR
002000                  AND QQ - Z - 3 (XY XZ) IS NOT  = ZERO
002001                      MOVE QQ - Z - 1 (XY XZ) TO CODE - IS
002002                      SET  XY  TO 11
002003                      SET  XZ  TO 11.
002004 *
002005  LOAD - ITEM SECTION.
```

```
002006 L - I - 1.
002007         PERFORM L - I - 2 VARYING XZ FROM 1 BY 1
002008              UNTIL XZ IS    MAX - Z.
002009         GO TO L - I - EXIT.
002010 L - I - 2.
002011         MOVE 'A' TO QQ - Z - I (XY, XZ)
002012         SET QQ - Z - 2 (XY, XZ) TO XY.
002013         SET QQ - Z - 3 (XY, XZ) TO XZ.
002014 L - I - EXIT.    EXIT.
002015 NXT - SEC SECTION.
002016 N - S - ALLL.
```

٦ ـ ٧ ـ ٢ البحث الثنائي BINARY SEARCH

البحث الثنائي يعتبر طريقة فعالة وسريعة في الجدول عن البند المطلوب و يتم في تعليمة البحث SEARCH آلياً مع اضافة كلمة ALL أي استخدام الصيغة رقم (٢) من الشكل العام السابق ذكره. كما يشترط لتحققه إضافة جملة بعبارة OCCURS عند توصيف الجدول تدل على طريقة ترتيب هذا الجدول وفق المفتاح الذي يتضمنه تصاعدياً أو تنازلياً لأنها سوف تحدد مسار البحث الثنائي في الجدول. وتصبح عبارة OCCURS عندئذ على النحو التالي في شكلها العام.

```
(level - number) (identifier) OCCURS (integer) TIMES

[ { ASCENDING  }  KEY IS data - name - 1 [data - name - 2] ... ]
[ { DESCENDING }                                                ]

   [INDEXED BY  index - name - 1 [index - name - 2] ....]
```

وتجدر الاشارة الى أنه عند استخدام البحث الثنائي مع كلمة ALL لا توجد حاجة الى تحديد نقطة بداية البحث في الجدول بتعليمة SET للدليل وإنما تتم آلياً ضمن تعليمة SEARCH ذاتها. والشكل التالي يوضح مثالا عن استخدام هذه التعليمة.

البحث باستخدام ALL بتعليمة SEARCH

(البحث الثنائي BINARY)

```
                    9.15.45          DEC 25, 1982
001530/
001540 01 TT - X.
001550     05 TT - Y OCCURS 5 TIMES
001560             ASCENDING KEY IS TT - Y - 2
001570             INDEXED BY TT - NDX.
001580          15 TT - Y - 1     PIC X.
001590          15 TT - Y - 2     PIC S9999.
001700 *
002019          PERFORM LOAD - IT VARYING TT - NDX FROM 1 BY 1
002020             UNTIL TT - NDX IS > 5.
002021          GO TO FIG - SEC.
002022 *     ........
002027 LOAD - IT SECTION.
002028 LD - 1.
002029          MOVE    'X'    TO TT - Y - 1 (TT - NDX).
002030          SET TT - Y - 2 (TT - NDX) TO TT - NDX.
002031 FIG - SEC SECTION.
002032 FIG - 1.
```

```
002034           SEARCH ALL TT - Y
002035               AT END GO TO ERR - NO - TT
002036               WHEN TT - Y - 2 (TT - NDX)    IS  = 4
002039               GO TO GOOD - TT - FOUND.
002040  ERR - NO - TT.
002041           DISPLAY 'NO ITEM FOUND ON TT - X TABLE'
002042                   UPON CONSLE.
002043           GO TO E - SEARCH - ALL.
002044 *
002045  GOOD - TT - FOUND.
002046           DISPLAY 'ITEM ON TT - X TABLE IS FOUND'
002047               UPON CONSOLE.
002048  E - SEARCH - ALL.
002049 *        .....
002050 *          ....
```

أسئلة وتمارين

سؤال ١ - لدينا الجدول المبين فيما يلي والذي يدعى SALARY - TAB والمطلوب توصيفه في قسم البيانات باستخدام عبارة OCCURS

SALARY - DATA (1)		SALARY - DATA (2)		SALARY - DATA (3)	
SALARY (1)	DATE (1)	SALARY (2)	DATE (2)	SALARY (3)	DATE (3)

سؤال ٢ - لدينا الجدول التالي واسمه SAL - TAB والمطلوب توصيفه في قسم البيانات باستخدام عبارة (أو عبارات) OCCURS مع جملة INDEXED BY .

SALARY - DATA					
SALARY (1)	SALARY (2)	SALARY (3)	DATE (1)	DATE (2)	DATE (3)

سؤال ٣ - في التوصيفين التاليين لجدولين بيّن حجم الجدول لكل منهما بعدد البايت؟ ثم وضح أي من التوصيفين يعتبر أكثر كفاية لأجل تداول الجدول .

```
1) ϕ1    STATE - TABLE.
      ϕ5    STATE - NAME OCCURS 50 TIMES       PIC    A (15).
      ϕ5    STATE - PPLATION OCCURS 5ϕ TIMES      PIC  9 (8).
```

```
2) ϕ1    STATE - TABLE.
    ϕ5    NAME - PPLATION OCCURS 50 TIMES INDEXED BY XST.
          1ϕ    STATE - NAME       PIC    A (15).
          1ϕ    STATE - PPLATION      PIC    9 (8).
```

حجم الجدول (1) .... بايت

حجم الجدول (2) .... بايت

الكفاية في التداول تعتبر أكبر : في التوصيف (1)

في التوصيف (2)

سؤال ٤ ـ لـدينا الجزء التالي من برنامج يحتوي على توصيف لجدول باسماء المدن في المملكة وهناك تعليمة بحث SEARCH في قسم الاجراءات من أجل تحديد اسم المدينة بعد معرفة رمزها من ملف القراءة.

```
01 TABLE - AREA.
    03   FILLER   PIC   X (15)   VALUE   '01 RIYADH '.
    03   FILLER   PIC   X (15)   VALUE   '02 JEDDAH '.
    03   FILLER   PIC   X (15)   VALUE   '03 DAMMAM '.
    03   FILLER   PIC   X (15)   VALUE   '04 QUASEEM'.
    03   FILLER   PIC   X (15)   VALUE   '05 HAYEL '.
    03   FILLER   PIC   X (15)   VALUE   '06 ALMADINA '.
    03   FILLER   PIC   X (15)   VALUE   '07 ABHA'.
    03   FILLER   PIC   X (15)   VALUE   '08 BALGURASHI'.
    03   FILLER   PIC   X (15)   VALUE   '09 ASEEF'.
    03   FILLER   PIC   X (15)   VALUE   '10 SHAQRA'.
*
01   RED - TABLE - AREA REDEFINES TABLE - AREA.
    05   LOCATIONS OCCURS 10 TIMES INDEXED BY LOC - INDEX.
        10   LOC - CODE    PIC   99.
        10   LOC - NAME    PIC   X (13).
*
PROCEDURE DIVISION.
.....
 ......
SET LOC - INDEX TO 1.
SEARCH LOCATIONS AT END
        MOVE 'UNKOWN' TO W - LOCATION
```

```
WHEN R - LOCATION  =  LOC - CODE (LOC - INDEX)
MOVE LOC - NAME (LOC - INDEX) TO W - LOCATION.
MOVE W - LOCATION TO P - LOCATION.
```

والمطلوب :

- بيان امكانية استخدام تعليمة البحث SEARCH مع كلمة ALL الاختيارية على هذا الجدول ؟ ولماذا اذا كان الجواب بالنفي ؟
- إعادة كتابة هذا الجزء بدون استخدام تعليمة SEARCH بحيث يؤدي نفس الغرض.

سؤال ٥ - لدينا توصيف الجدول في قسم البيانات كما يلي :

```
1)   ϕ1        PAYMENTS
        ϕ5   FILLER   PIC   X (8)   VALUE   "17161819".
        ϕ5   FILLER   PIC   X (8)   VALUE   "1456155ϕ".
        ϕ5   FILLER   PIC   X (8)   VALUE   "12143748".
     ϕ1        PAYMENTS - TAB REDEFINES PAYMENTS.
        ϕ5   STARTING - PERIOD OCCURS 3 TIMES INDEXED BY
                                S - INDEX.
          1ϕ   ENDING - PERIOD OCCURS 2 TIMES INDEXED BY
                                E - INDEX.
            15   PAYMENT PIC 99V99.
```

وفي قسم الاجراءات التعليمات التالية :

```
2)   SET   S - INDEX TO 3.
     SET   E - INDEX TO 1.
```

```
MOVE    PAYMENT (S - INDEX,    E - INDEX) TO PAYMENT - OUT.
SET     S - INDEX DOWN BY 1.
MOVE    PAYMENT (S - INDEX,    E - INDEX) TO PAYMENT - LIST.
```

والمطلوب :

١ ـ رسم توضيحي لشكل الجدول الموصوف PAYMENTS على شكل صفوف وأعمدة.

٢ ـ بيان قيمة الحقل PAYMENT - OUT وقيمة الحقل PAYMENT - LIST. بعد تنفيذ التعليمات المذكورة سابقاً في قسم الاجراءات علماً بأن شكل الحقلين هو 99.99.

تمرين ١ ـ أكتب البرنامج التالي :

لدينا الجدول التالي والذي يمثل قيمة مبيعات إحدى الشركات لثلاث سلع مختلفة وعلى مدى أيام الأسبوع الستة. (المبالغ بألوف الريالات) :

DAYS	GOODS		
	FOOD	CLOTHES	EQPMT
SAT	1ɸ	2ɸ	3ɸ
SUN	ɸ2	ɸ3	04
MON	15	16	17
TUE	4ɸ	41	42
WED	ɸ8	ɸ9	1ɸ
THR	5ɸ	51	52
TOTAL	???	???	???

المطلوب :

أ ـ توصيف هذا الجدول في قسم البيانات من أجل إجراء عمليات الجمع والطباعة.

ب ـ كتابة الاجراءات اللازمة لجمع قيمة السلع GOODS الثلاث في أيام الاسبوع كما هو مبين في الجدول.

جـ ـ كتابة التعليمات والاجراءات اللازمة لطباعة هذا الجدول على الشكل التالي :

DAY	FOOD	CLOTHES	EQUIPMENT
XXX	999	999	999
.	.	.	.
.	.	.	.
.	.	.	.
TOTAL	999	999	999

ملاحظة : استخدم الفراغات وأسماء الحقول المناسبة واستعن برسم FLOWCHART

الفصل السابع

الفـــــرز والدمـــــج
SORT / MERGE

٧ ـ ١ لمحة عامة

من أجل زيادة السرعة في معالجة الملفات التسلسلية Sequential التي تحتوي على حقل أو أكثر يعتبر بمثابة مفتاح Key للسجل الموجود في الملف، أصبح من الواجب فرز (Sort) هذه الملفات طبقاً لتسلسل معين في هذا الحقل : أما تنازلي Descending أو تصاعدي Ascending .

وعملية الفرز يمكن أن تتم باحدى الطرق الثلاث التالية :

١ ـ الفرز بوساطة إجراء معين يكتبه المبرمج في برنامجه أو في برنامج خاص وهناك طرق عدة واساليب مختلفة لإجراء الفرز لا مجال لذكرها هنا.

٢ ـ الفرز بوساطة برنامج جاهز خارج برنامج كوبول يدعى أحياناً Sort Utility وهذا البرنامج الجاهز يختلف من شركة صانعة الى أخرى (من حاسب الى آخر) و يعطي المبرمج أو المشغل الفرصة بادخال المعطيات الخاصة Parameter بمتطلبات عملية الفرز مثل اسم الملف المراد فرزه وطول السجل واسم الحقل الذي يتم الفرز على أساسه (Major Key) واسماء الحقول الأخرى (Minor Keys) واسم الملف الذي يحتوي نتيجة الفرز ومعلومات أخرى تتعلق بعملية الفرز ولا داعي للدخول في تفاصيله لأنه يختلف باختلاف نوع الحاسب ولكن من خواصه أنه سهل التطبيق وسريع التنفيذ.

٣ ـ الفرز بوساطة تعليمات خاصة تذكر في داخل برنامج كوبول تؤدي الى تنفيذ عملية فرز الملف وثم متابعة تنفيذ البرنامج الذي يحتوي تعليمات الفرز، وهذا الاسلوب الذي يهم المبرمج بلغة كوبول معرفته في هذا المؤلف.

٧ ـ ٢ الفرز في لغة كوبول SORT

بما أن عملية الفرز مطلوبة غالباً للمحافظة على نظام تسلسلي للبيانات فان لغة كوبول قد طورت لتشمل (بعد عام ١٩٧٤) ميزة الفرز هذه وتوفر على المبرمج في الدخول في تفاصيل الطرق التقنية للفرز ولتكون بعيدة عن التأثر بنوع الحاسب المستخدم ومستقلة عن برامجه الجاهزة. ولتوضيح عملية الفرز نأخذ المثال التالي :

لنفترض أنه يوجد لدينا ملف على شريط ممغنط (أو على قرص ممغنط رتب على أساس تسلسلي) يحتوي على وصف التالي في قسم البيانات Data Division :

```
ϕ1   INTAPE - REC.
     ϕ5   STDNT - NO        PIC 9 (7).
     ϕ5   STDNT - NAME        PIC X (3ϕ).
     ϕ5   STDNT - MRKS.
          ϕ7   SCORE        PIC 99V99.
          ϕ7   GRADE         PIC X.
     ϕ5   FILLER        PIC X (38).
```

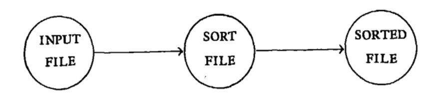
ونرغب في فرز هذا الملف تصاعدياً وفق رقم الطالب STDNT - NO وتنازلياً وفق العلامة SCORE في آن واحد. ونفترض بأن الملف المفروز يدعى SORTED - FILE

وسوف يحفظ على وحدة تداول ثالثة . فيمكن تصور عملية الفرز والتي سوف تشتمل على ثلاثة ملفات :

INPUT - FILE

و SORT - FILE

و SORTED - FILE



لأن عملية الفرز تحتاج الى ملف (احياناً أكثر من ملف) ليستخدم بمثابة منطقة عمل خاصة بالفرز (SORT FILE) .

والشكل المبين في الصفحة التالية يمثل برنامج كوبول يحتوي على عملية فرز للملف الموصوف في الفقرات السابقة والذي يحتوي على معلومات عن الطلاب و يتم الفرز وفق رقم الطالب تصاعدياً ووفق علامته تنازلياً .

و يلاحظ في الشكل أنه تم توصيف ثلاثة ملفات في قسم المحيطات ENVIRONMENT DIVISION بعبارة SELECT وكذلك في قسم توصيف البيانات DATA DIVISION ثم توصيف ملف الادخال INPUT - FILE وملف الفرز SORT - FILE وملف الاخراج SORTED - FILE . وتستخدم عبارة SD للملف الفرز في التوصيف و يقصد بها Sort file Description بدلا من FD المستخدمة في الملفات الأخرى . كما يلاحظ في هذا القسم عدم وجود عبارة ... LABEL RECORD للملف الفرز (رغم وجود بعض المترجمات التي تسمح بها) لأن ANSI COBOL اعتبرت ملف الفرز كمنطقة تخزين وعمل داخلي في البرنامج .

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID.    SORTP.
 AUTHOR. AJND-KHAYAT.
**       THIS PROGRAM IS TO ILLUSTRATE  **
**          COBOL   SORT STATEMENT.     **
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER. IBM-3033.
 OBJECT-COMPUTER. IBM-3033.
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
        SELECT INPUT-FILE ASSIGN TO
                                  UT-S-SYSIN.
        SELECT SORT-WORK-FILE ASSIGN TO
                              UT-S-WORKSORT.
        SELECT SORTED-FILE ASSIGN TO
                              UT-S-SORTOUT.
 DATA DIVISION.
 FILE SECTION.
 FD    INPUT-FILE  LABEL RECORD IS STANDARD
                  DATA RECORD IS INPUT-REC.
 01    INPUT-REC.
       05  STD-NO                  PIC 9(7).
       05  STD-NAME                PIC X(30).
       05  STD-MARK.
           07  STD-SCORE           PIC 99V99.
           07  STD-GRADE           PIC X.
       05  FILLER                  PIC X(38).
*
 SD    SORT-WORK-FILE
            RECORD CONTAINS  80 CHARACTERS
                  DATA RECORD IS SORT-REC.
 01    SORT-REC.
       05  STD-NO-S                PIC 9(7).
       05  STD-NAME-S              PIC X(30).
       05  STD-MARK-S.
           07  STD-SCORE-S         PIC 99V99.
           07  STD-GRADE-S         PIC X.
       05  FILLER                  PIC X(38).
 FD    SORTED-FILE
                 LABEL RECORD IS STANDARD
```

```
                   DATA RECORD IS SORTED-REC.
 01    SORTED-REC                PIC X(80).
*
 PROCEDURE DIVISION.
 SORT-ROUTINE.
      SORT  SORT-WORK-FILE
          ON  ASCENDING KEY STD-NAME-S
          ON  DESCENDING KEY STD-SCORE-S
         USING     INPUT-FILE
         GIVING    SORTED-FILE.
        STOP RUN.
```

```
**
***     UNSORTED  FILE                 **********
0234517AHAMD                                7050C
0234527ZAID                                 8050B
0234537ALI                                  9050A
0234547WALEED                               5050F
0234557FAROUQ                               6050D
0234557HISHAM                               6050D
0234557JAMEEL                               6650D
0234557ABED-ALAZIZ                          9650A
0234557MAZEN                                8650A
0234557NABEEL                               7650A
**
***      SORTED  FILE              *************
0234557ABED-ALAZIZ                          9650A
0234517AHAMD                                7050C
0234537ALI                                  9050A
0234557FAROUQ                               6050D
0234557HISHAM                               6050D
0234557JAMEEL                               6650D
0234557MAZEN                                8650A
0234557NABEEL                               7650A
0234547WALEED                               5050F
0234527ZAID                                 8050B
```

أما قسم الاجراءات PROCEDURE DIVISION فهو بسيط و يشتمل على فقرة واحدة، وهو أمر الفرز SORT وهذا الأمر (أو التعليمة) فعال ومريح للمبرمج حيث أنه لا يحتاج الى توصيف معقد لأدلة الفرز Keys ومصدر البيانات (الادخال) ووجهة البيانات (الاخراج)، بل و يُكتفى بذكر بسيط وسهل لهذه المعطيات.

وفي حال كون المبرمج بحاجة الى عملية الفرز على أكثر من دليل* (مفتاح) فيمكنه إضافة اسم المفتاح بعد كلمة ASCENDING أو DESCENDING و يعتبر الاسم الأول بمثابة مفتاح كبير Major Key والتالي له مفتاح أصغر يدعى عادة Minor Key . وعدد هذه المفاتيح في التعليمة يقارب سبعة مفاتيح ويختلف من مترجم كوبول الى آخر وامكانية نظام التشغيل المتوفرة.

وعبارة USING INPUT - FILE تحدد الملف الذي يمثل مصدر البيانات المراد فرزها بينما عبارة GIVING SORTED - FILE تحدد وجهة البيانات بعد الفرز.

و يلاحظ في الشكل السابق أيضاً أن عملية فتح وإغلاق CLOSE ، OPEN للملفات لم تذكر صراحة في البرنامج لأن تعليمة SORT تتضمن ذلك آلياً.

الشكل العام لتعليمة الفرز SORT

فيما يلي عرض للأشكال العامة للتوصيفات اللازمة لتعليمة الفرز في أقسام البرنامج المختلفة.

ففي قسم التعريف IDENTIFICATION لا يوجد أي استثناء عن التوصيف العادي.

(*) دليل أو مفتاح الفرز Keys

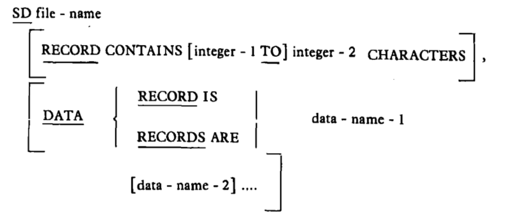
أما في قسم المحيطيات ENVIRONMENT وفي الجزء الخاص بالملفات تأخذ عبارة SELECT الشكل التالي :

```
FILE - CONTROL.

        SELECT file - name ASSIGN TO device - name - 1
            [device - name - 2 ]..... [OR device - name - 3]
            [device - name - 4] ...
        [                      { REEL }  ]
        [  FOR MULTIPLE        {      }  ]  ....
        [                      { UNIT }  ]
```

ووجود كلمة OR في هذه العبارة تسمح باخراج نتيجة الفرز على احدى الوساطات (وحدات تداول التخزين الخارجي) المعرفة بعد هذه الكلمة.

أما قسم البيانات DATA فيمكن أن يظهر على الشكل التالي :



```
SD file - name
   [RECORD CONTAINS [integer - 1 TO] integer - 2  CHARACTERS] ,
   [          { RECORD IS    }                              
   [ DATA     {              }        data - name - 1       
   [          { RECORDS ARE  }                              
                  [data - name - 2] ....                    ]
```

حيث أن كلمة SD تدل على وصف لملف الفرز والعبارة التالية RECORD CONTAINS تحدد طول السجل RECORD الذي يحتوي ملف الفرز ويحسب بالبايت Bytes . و يلاحظ هنا أن عبارة BLOCK CONTAINS الاختيارية لم تذكر لأن حجم الشكل Blocks سوف يقوم روتين الفرز بتحديده آلياً .

أما عبـارة DATA و RECORD أو RECORDS فتصف اسم (أو أسماء) السجل (السجلات) الموصوف (الموصوفة) تحت اسم ملف الفرز.

أما قسم الاجراءات PROCEDURE فيمكن أن يحتوي تعليمة الفرز التي تأخذ الشكل العام التالي مع عباراتها الاختيارية :

```
SORT    file - name - 1

    ON { ASCENDING  } KEY data - name - 1 [,data - name - 2]
       { DESCENDING }

  [ ON { ASCENDING  } KEY data - name - 3[,data - name - 4]... ]
  [    { DESCENDING }                                          ]

        [COLLATING SEQUENCE IS alphabet - name]

{ INPUT PROCEDURE IS Section - name - 1 [ { THRU    } Sect - name - 2 ] }
{                                       [ { THROUGH }                 ] }
{ USING file - name - 2 [, file - name - 3] ....                        }

{ OUTPUT PROCEDURE IS Section - name - 3 [ { THRU    } Sect - name - 4 ] }
{                                        [ { THROUGH }                 ] }
{ GIVING file - name - 4                                                 }
```

حيـث أن file - name - 1 بعد فعل الفرز SORT يقصد به ملف الفرز الموصوف في عبـارة SD . كـمـا أن data - name - 4 و data - name - 2 و data - name - 1 هـي أسـمـاء الحـقـول المـعـرفـة في مـلـف الـفـرز والـتـي تـعـتبر بمثابة مفاتيح فرز. أما كلمة

ASCENDING تستخدم للفرز تصاعدياً و DESCENDING للفرز تنازلياً بالنسبة لاسم المفتاح الذي يليهما . فمثلا :

SORT file - name ON ASCENDING KEY
YEAR, STDNT - NAME, STDNT - NO

وهنا يتم الفرز تصاعدياً وفق السنة YEAR ثم اسم الطالب STDNT - NAME ثم رقم الطالب STNDT - NO داخل الاسم وداخل السنة . وهذا الشكل يعادل :

```
SORT file - name ON ASCENDING KEY YEAR
          ON ASCENDING KEY STDNT - NAME
          ON ASCENDING KEY STDNT - NO
```

وعبارتا INPUT PROCEDURE و OUTPUT PROCEDURE تشيران الى اسم القسم (Section) أو مجموعة الفصول * عند استخدام كلمة THRU.

والتعليمات التي يمكن أن ترد في فصل أو فصول عبارة INPUT PROCEDURE تقوم باجراء بعض المعالجة على الحقول أو السجلات الواردة من ملف الإدخال مثل كتابتها على ملف الفرز لأجل فرزها، وغالباً ما تكون هذه المعالجة مؤقتة وتحويلية . وفي هذه الحالة يجب على المبرمج القيام بفتح الملف (ملف الادخال) واستعمال تعليمة القراءة (READ) وثم معالجته . ولأجل الكتابة على ملف الفرز تستخدم تعليمة RELEASE والتي تعادل تعليمة WRITE الوارد ذكرها في الفصول السابقة . وتأخذ تعليمة RELEASE الشكل العام التالي :

__RELEASE__ record - name [__FROM__ identifier]

(*) الفصل هو مجموعة فقرات من قسم الاجراءات

حيث يدل record - name على اسم سجل ملف الفرز. وتساعد هذه التعليمة في نقل محتويات ملف الادخال الى ملف الفرز.

وكذلك الحال في عبارة OUTPUT PROCEDURE التي تحتوي تعليمات خاصة بمعالجة تحويلية أو مؤقتة للسجلات بعد قراءتها من ملف الفرز وارسالها الى ملف الاخراج. وتستخدم هنا تعليمة خاصة بالقراءة من ملف الفرز RETURN وتأخذ الشكل العام التالي :

RETURN file - name RECORD [INTO identifier]

AT END imperative statement

حيث تمثل عبارة file - name اسم ملف الفرز. وعند استخدام عبارة OUTPUT PROCEDURE يجب على المبرمج مراعاة فتح ملف الاخراج وإغلاقه.

أما عبارتا USING و GIVING فهما تستخدمان عندما لا توجد اجراءات خاصة عند ادخال البيانات الى ملف الفرز أو عند إخراجها منه. و يأتي اسم ملف الإدخال بعد كلمة USING وأحياناً يكون أكثر من ملف. أما اسم ملف الاخراج (أو وجهة البيانات بعد الفرز) فيذكر بعد كلمة GIVING.

وأخيراً تعطي عبارة COLLATING SEQUENCE المبرمج فرصة لتحديد تسلسل (ترتيب) أبجدي معين للبيانات المراد فرزها، وقد يختلف هذا الترتيب عن الترتيب القياسي المعروف باسم EBCDIC أو باسم ASCII ويحدد اسم الترتيب تحت اسم الفقرة الخاصة التالية :

SPECIAL - NAMES

في قسم المحيطيات.

٧ ـ ٣ دمج الملفات FILE MERGING

إن دمج الملفات إجراء شائع في معالجة البيانات والدمج يعني ببساطة جمع ملفين أو أكثر في ملف واحد. فمثلاً اذا كان هناك مصرف يوجد فيه ملف على شريط ممغنط للعمليات اليومية مرتبة حسب رقم الزبون و بشكل يومي عندئذ تتطلب المعالجة جمع هذه الملفات اليومية (المفرزة وفق رقم الزبون) في ملف واحد شهري بحيث تظهر كافة العمليات مرتبة مع بعضها لكل زبون أو عميل.

ولأن المبرمج يحتاج اليه تكراراً ولأنه ذو أهمية كبيرة فقد طبق في لغة كوبول باعتباره ميزة في لغة عالية جداً، وأخذ هذا التطبيق شكل تعليمة تسمى الدمج MERGE.

ولنأخذ مثال على ذلك أن شركة أعمال تنتج (تخرج) في نهاية كل ربع سنة مواداً وتم تسجيل بيانات هذه المواد على ملف وكل سجل في هذا الملف يحتوي على رقم القسم ورقم المادة المباعة بالاضافة الى بيانات أخرى. وهذا الملف فُرز (رتُب) حسب رقم القسم كمفتاح أساسي Major Key ورقم المادة (المنتج) المباعة كمفتاح فرعي Minor في عملية الفرز. وفي نهاية العام الشركة مهتمة بدمج هذه الملفات الأربعة الربعية وجمعها في ملف واحد سنوي.

والشكل التالي يوضح الاطار العام للاجزاء اللازمة في برنامج كوبول :

```
FD      FIRST - QRTR      LABEL RECORD STANDARD
                          DATA RECORD SALES - HSTRY.
ϕ1      SALES - HSTRY.
        ϕ5    DEPT - NO        PIC    99.
```

```
        ϕ5    PRDUCT - NO          PIC    9 (6).
        |
        |
FD    SECOND - QRTR ...
        |
        |
FD    THIRD - QRTR ...
        |
        |
FD    FOURTH - QRTR ...
        |
        |
FD    YEARLY - FL    LABEL RECORD STANDARD

                     DATA RECORD YR - REC.

01    YR - REC.

        ϕ5    YDEPT - NO          PIC    99.

        ϕ5 YPRDUCT - NO           PIC    9 (6).
           |
           |
SD    MERGE - FILE          DATA RECORD MERGE - REC.

ϕ1    MERGE - REC.

        ϕ5    MDEP - NO          PIC    99.

        ϕ5    MPRD - NO          PIC    9 (6).
        |
        |
PROCEDURE    DIVISION.
   |
   |
MERGE        MERGE - FILE ON ASCENDING KEY MDEP - NO

                          ON ASCENDING KEY MARD - NO

             USING   FIRST - QRTR, SECOND - QRTR,

                     THIRD - QRTR, FOURTH - QRTR

             GIVING YEARLY - FL.
```

يلاحظ من الشكل السابق أن هناك أربعة ملفات موصوفة بجملة FD ثم هناك ملف موصوف بجملة SD ليستخدم في عملية الدمج والذي اسمه MERGE - FILE . و يلاحظ تطابق تفصيلات السجل في هذا الملف مع السجل في الملفات الأخرى .

وفي قسم الاجراءات تشير عبارة MERGE الى الملف الموصوف بـ SD والذي يحدد الدمج على أساس المفتاح الأساسي (رقم القسم) MDEP - NO والمفتاح الذي هو رقم المادة MPRD - NO .

وجملة USING تحدد الملفات التي سوف تدمج والتي تعتبر ملفات إدخال . وهذه الملفات يجب أن تكون مغلقة CLOSED عند عملية الدمج لأن فتح الملف بوساطة MERGE يتم ضمنياً .

وجملة GIVING تحدد الملف الناتج من عملية الدمج وهو الذي يحتوي على الملفات الأ ربعة الربع سنوية . وهذا الملف الجديد سوف يكون بنفس الترتيب SORT الذي كانت عليه الملفات الربع سنوية .

والشكل العام (الصيغة العامة) لتعليمة MERGE كما يلي :

```
MERGE   file - name - 1

     ON  { ASCENDING  }  KEY data - name - 1 [, data - name - 2] ...
         { DESCENDING }

   [ ON  { ASCENDING  }  KEY data - name - 3 [, data - name - 4]... ] ...
   [     { DESCENDING }                                              ]
```

[COLLATING SEQUENCE IS alphabet - name]

USING file - name - 2, file - name - 3 [, file - name - 4]

```
{ OUTPUT PROCEDURE IS section - name - 1 [ { THROUGH } section - name - 2 ] }
{                                          [ { THRU    }                   ] }
{ GIVING file - name - 5                                                     }
```

وعبارة OUTPUT PROCEDURE تعادل في استخدامها نفس العبارة المذكورة مع تعليمة SORT . وكذلك تستخدم عبارة RETURN في روتين إجراء الإخراج لجعل السجلات المدموجة جاهزة للمعالجة، تماماً كما في حالة الـ SORT . ولكن بعكس حالة SORT فان عبارة MERGE لا تشمل أي اختيارات تتعلق بروتين إجراء الإدخال، لأن ملفات الإدخال يجب أن تكون في شكل ملائم ومناسب لعملية الدمج مثل تنفيذ عبارة MERGE .

٧ ـ ٤ اسئلة وتمرينات

سؤال ١ ـ بين الصحيح والخطأ من العبارات التالية :

- إذا ذكرت عبارة USING في تعليمة SORT فيجب ذكر عبارة GIVING معها .
- إذا تم ذكر عبارة INPUT PROCEDURE مع تعليمة SORT عندئذ يجب ذكر عبارة OUTPUT PROCEDURE
- يسمح بذكر مفتاح تصاعدي واحد أو مفتاح تنازلي واحد في تعليمة SORT .

- تستخدم تعليمة RELEASE في فقرات INPUT PROCEDURE .
- تستخدم تعليمة RETURN في فقرات OUTPUT PROCEDURE .
- إذا استخدمت عبارتا USING / GIVING مع فصل الفرز فان الملف المفروز SORTED FILE يجب أن يحتوي على كل سجل من ملف الإدخال INPUT FILE .
- يوصف اسم ملف الفرز وملف الدمج في قسم البيانات في فصل الملفات بعبارة SD .
- لا تقبل تعليمة MERGE أربعة ملفات كملفات إدخال .
- تستخدم تعليمة MERGE لدمج ملف أو أكثر من ملف واحد.

سؤال ٢ ـ لديك أسماء الحقول التالية مع بياناتها :

الاسم	الجهة	القسم
محمد	الرياض	١٠٠٠
سامي	أبها	٢٠٠٠
ممدوح	أبها	٢٠٠٠
ياسر	جدة	٣٠٠٠
عبدالله	القصيم	٤٠٠٠
عبدالحميد	الرياض	١٠٠٠
عبدالرحمن	الرياض	١٠٠٠
بخيت	الدمام	٥٠٠٠

اعد ترتيب البيانات السابقة وفق ما يلي :

١ ـ المفتاح الأولي MAJOR KEY هو حقل القسم (تنازلياً)

٢ ـ المفتاح الثانوي MINOR KEY هو حقل الاسم (تصاعدياً)

سؤال ٣ - في العبارة التالية :

```
SORT  SORT - FILE
        ASCENDING KEY STD - NAME
        DESCNDING KEY YEAR - IN - SCHOOL
        ASCENDING MAJOR
        USING FILE - ONE
     GIVING FILE - TWO.
```

أجب على الاسئلة التالية :

أ - ما هو المفتاح الأساسي ؟

ب - ما هو المفتاح الثانوي ؟

جـ - أي من الملفات السابقة سيوصف تحت عبارة SD ؟

د - أي من الملفات سوف يوصف في عبارة SELECT ؟

هـ - أي ملف سيكون ملف إدخال ؟

و - أي من الملفات يجب أن يحتوي على اسماء الحقول YEAR - IN - SCHOOL - STD - NAME, و MAJOR ؟

تمرين ١ :

ثلاثة ملفات تحوي على معلومات عن سجلات زبائن أحد المصارف موجودة على قرص ممغنط ، المطلوب كتابة برنامج يجري العمليات التالية :

١ - دمج MERGE هذه الملفات الثلاثة في ملف واحد مفروز تصاعدياً على رقم الزبون ووضع الناتج على ملف اخراج اسمه OTMRG .

٢ - طباعة تقرير من ملف OTMRG يحوي على رقم الزبون واسمه والرصيد على أن

يطبع في نهاية العمل عدد الزبائن ومجموع الأرصدة.

٣ - إذا كان الرصيد سالباً أطبع عبارة «*******» في حقل الملاحظات علماً بأن شكل السجل في الملفات الأربعة هو:

	FIELD NAME	PICTURE
رقم الزبون	CUST - NO	9 (6)
اسم الزبون	CUST - NAME	X (24)
الرصيد	CUST - BAL	S9 (8) V99
غير مستعمل	CUST - DATA	X (4ϕ)

* اختر اسماء الملفات والحقول المناسبة.

** اطبع في الصفحة ٣٠ سطراً فقط. واترك سطر فراغ بين كل سطري طباعة - انظر شكل توصيف المخرجات صفحة ٦٠٥.

*** أظهر الاشارة الجبرية في حقل الرصيد.

تمرين ٢ :

ملف يحتوي على سجلات سلع لاحدى المخازن اسمه STOCKF موجود على قرص ممغنط، وشكل السجل فيه كالتالي:

	FIELD NAME	PICTURE
رقم السلعة	STK - NO	9 (5)
وصفها	STK - DESC	X (2ϕ)
الكمية	STK - QTY	9 (5)
رمز السلعة	STK - CODE	9 (3)
سعر الوحدة	STK - PRC	9 (3)V99
حد الطلب	STK - MIN	9 (5)
غير مستعمل	FILLER	X (37)

هـذا المـلـف STOCKF تـرتـيـب الـسلع فيه ترتيباً عشوائياً. المطلوب كتابة برنامج يجري العمليات التالية :

١ - فـرز SORT هـذا المـلـف تـصاعدياً على رقم السلعة STK - NO وأن يوضع ناتج عـمـلـيـة الـفـرز على مـلـف آخـر اسـمـه SRTSTK بـه نفس شكل وطول ملف الإدخال.

٢ - طـبـاعة الملف SRTSTK على الطابعة بشكل تقرير يظهر رقم السلعة، ووصفها، والكمية، ورمز السلعة، وسعر الوحدة.

٣ - مـقـارنـة الكمية STK - QTY مع حد الطلب STK - MIN فإذا كانت الكمية مـسـاويـة أو اصـغـر لحـد الطلب اطبع عبارة أعد الطلب 'REORDER' في حقل الملاحظات - انظر شكل توصيف المخرجات صفحة ٦٠٦ .

ملاحظات :

١ - اطبع في الصفحة الواحدة ٢٥ سلعة فقط.

٢ - اطبع في رأس كل صفحة العناوين والتاريخ ورقم الصفحة.

٣ - اترك فراغاً بين كل سطري طباعة.

الفصل الثامن

الملفـــــــــــات

FILES

ومعـــالجـــــــتها

٨ ـ ١ لمحة عامة

نعلم إنه في أي نظام يدوي يتم تخزين البيانات DATA في ملفات ورقية ـ أو بلاستيكية ـ و يكتب عليها معلومات أساسية مثل محتوى هذا الملف، رقم الملف، تاريخه والى ما هنالك من معلومات تخص كل نظام.

أما عندما نريد تحويل النظام اليدوي CLERICAL SYSTEM الى نظام آلي ينفذ عن طريق الحاسب، فنكون بحاجة الى ملفات على الحاسب تخزن البيانات المختلفة للنظام. ويمكن معالجتها والرجوع إليها بسهولة باستخدام برامج الحاسب، هذه الملفات الخاصة بالحاسب لها أنواع و بنية وتنظيم وأساليب معالجة تختلف عن الملفات اليدو ية ـ باختلاف وسط التخزين طبعاً ـ لتوفر السرعة في استدعاء البيانات والدقة في معالجة هذه البيانات، واختصار حجم التخزين طبعاً عن الملفات اليدو ية.

٨ ـ ٢ أنواع الملفات TYPE OF FILES*

يمكن تقسيم أنواع الملفات حسب طبيعة البيانات التي تحتو يها الى الأنواع التالية :

٨ ـ ٢ ـ ١ ملفات أساسية MASTER FILES :

وهذه من أهم أنواع الملفات والتي يرتكز عليها أي نظام، لما تحتو يه من بيانات

(*) N.C.C. INTRODUCING SYSTEMS ANALYSIS AND DESIGN VOL 2 1980 PP. 179 - 209.

أساسية، هذه البيانات تتكرر دوماً في أي دورة تشغيل للنظام. وهنالك نوعان من الملفات الأساسية :

٨ - ٢ - ١ - ١ ملفات مصدرية REFERENCE FILES :

وتحتوي على بيانات لا تتغير بصورة دورية مثل اسم الموظف، عنوانه، تاريخ ميلاده، رقم حفيظة النفوس (جواز سفره)، ... هذه المعلومات شبه ثابتة تقريباً.

٨ - ٢ - ١ - ٢ ملفات كثيرة التبدل DYNAMIC FILES :

وتحتوي على بيانات تتغير وفق احداث أو تعديلات عملية TRANSACTIONS معينة.

مثل ملف المخزون في نظام المستودعات. يظهر الكميات الجديدة المضافة الى المستودع، والكميات التي خرجت من المستودع والرصيد الحالي لكل صنف من أصناف المخزون.

كما أن ملف حجوزات السفر يظهر المسافرين الذين تم حجز أماكن لهم والوضع الحالي للأماكن الشاغرة.

ويمكننا أن نضيف الى أن النوعين السابقين من الملفات الأساسية المصدرية والمتبدلة يمكن أن يكونا منفصلين. ويمكن أن يجتمعا في ملف واحد مختلط يجمع بين مميزات الملفين، مثل نظام المحاسبة، فالسجل الواحد يمكن أن يجمع معلومات مصدرية مثل اسم الحساب، ورقم الحساب. ومعلومات خاصة بالحساب، كما يحتوي على المعلومات المالية المتغيرة. مثل الأرصدة لكل حساب.

٨ - ٢ - ٢ ملفات إدخال INPUT FILES :

تحمل هذه الملفات البيانات الى النظام من خلال تدقيقها ببرامج تحقق VALIDATION PROGRAM المكتوبة على الحاسب. اذ يتم ادخال البيانات عبر

بطاقات مثقبة PUNCHED CARDS أو عن طريق الشاشات TERMINALS الى هذه الملفات (تكون وقتها OUTPUT ملفات اخراج). بعدها تعالج هذه الملفات كملفات إدخال للنظام، وتمرر على برنامج تحقق يسجل البيانات الصحيحة على ملفات أساسية والبيانات غير الصحيحة على ملفات أخرى أو يطبع بهم تقرير يظهر الخطأ، ثم تصحح هذه الأخطاء وتتم معالجة الملفات من جديد.

٨ ـ ٢ ـ ٣ ملفات انتقالية TRANSFER FILES :

تحوي هذه الملفات على بيانات تساعد في مراحل متقدمة من تشغيل النظام مثل ملفات الطباعة أو تحديث ملف آخر UPDATE ويمكن أن تكون مفروزة ـ SORTED ـ حسب مفتاح ـ KEY ـ معين.

٨ ـ ٢ ـ ٤ ملفات عمل WORK FILES :

هذه الملفات هي ملفات وسيطية تحتوي على بيانات مختارة من ملف أو أكثر. وهذه البيانات المختارة تحتاج لمعالجة مؤقتة ولا يهتم النظام لحفظها لفترة طويلة مثل ملفات الفرز، SORT أو الملفات التي يُحتاج اليها لإجراء بعض العمليات الحسابية الطويلة وتخزينها مؤقتاً على ملفات خاصة.

٨ ـ ٢ ـ ٥ ملفات إخراج OUTPUT FILES :

هذه الملفات تحمل البيانات من الحاسب الى الطابعة مثلا أو لادخالها في نظام آلي آخر.

مثال :

الفواتير التي سترسل الى الزبائن تطبع على الطابعة (ملف اخراج) أما الفواتير التي ستدخل في النظام المحاسبي الآلي فانها تسجل على ملف إخراج، يكون فيما بعد ملف

ادخال في النظام المحاسبي، ولا ننسى أن هذه المخرجات هي مشتقة من ملف أو أكثر.

٨ ـ ٢ ـ ٦ ملفات الصورة الحقيقية DUMP FILES :

من أجل المحافظة على سلامة التشغيل والبيانات التي يقوم الحاسب بمعالجتها يجري عادة أخذ نسخة عن محتويات ذاكرة الحاسب بين الحين والآخر في فترة الشتغيل، وحفظها على صورتها الحقيقية. PHYSICAL في ملف من أجل الرجوع اليه عند الحاجة.

٨ ـ ٢ ـ ٧ ملفات مكتبة الحاسب *LIBRARY FILES :

تستخدم هذه الملفات لحفظ البرامج التطبيقية أو البرامج الجزئية MODULES PROGRAMS. أو برامج الخدمة UTILITY PROGRAMS. أو بعض برامج التشغيل SOFTWARE الذي يحتاجه الحاسب.

٨ ـ ٢ ـ ٨ ملفات تعديل ـ ثانوية ـ TRANSACTION FILES :

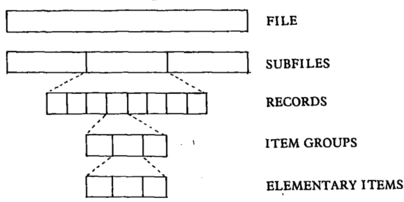
تحتوي هذه الملفات على بيانات خاصة تعدل الملف الأساسي وتكون مفروزة تصاعدياً أو تنازلياً على مفتاح يطابق في توصيفه مفتاح الملف الأساسي، وكذلك تحتوي أحياناً على حقل يسمى دليل العملية TRANSACTION CODE تتم وفقه مختلف أنواع العمليات على الملف الأساسي وهذه العمليات هي : حذف سجل تعديل محتوى سجل وإضافة سجل جديد.

(*) LIBRARY : هي عبارة عن مساحة على القرص الممغنط مخصصة لاحتواء البرامج ذات الاستخدام العام لكافة المستفيدين من الحاسب، وتحتوي على عدة برامج جاهزة من الشركة الصانعة و برامج مكتوبة من قبل ادارة تشغيل الحاسب وتكون ذات طبيعة مشتركة.

٨ ـ ٣ بنية الملف FILE STRUCTURE

يمكن تقسيم الملف الى عدة مستويات هي :

- يمكن أن يقسم الملف الى ملفات جزئية ـ عند استخدام رمز معين CODED RECORDS.
- كل ملف يقسم الى سجلات RECORDS.
- السجل يقسم الى حقول FIELDS وهو أصغر وحدة يمكن معالجتها ببرنامج الكوبول.
- الحقل يقسم الى أحرف CHARACTERS (رقمي أو أبجدي).



وفي الصفحات التالية شرح مكونات الملف (الملفات) *

٨ ـ ٣ ـ ١ الملفات الجزئية SUBFILES :

يمكن تقسيم بعض الملفات من الناحية المنطقية الى ملفات أصغر منها، فلو أخذنا على سبيل المثال ملف الحركة في المحاسبة. فهو يحتوي على مبالغ منه ومبالغ له يرمز لها برمز معين فلو فرضنا الرمز ١ يعني أن المبلغ مدين DEBIT وإذا كان الرمز ٢ يعني دائن

(*) N.C.C PAGE 181

CREDIT ، فنحن كأننا أمام ملفين أحدهما دائن والآخر مدين لكن سجلاتهما مدموجين في ملف واحد.

ومهما يكن من الأمر فليس المقصود من الملف الجزئي أنه يفتح بتعليمات لوحده و يعالج و يغلق بتعليمات مستقلة انما التقسيم من الناحية المنطقية LOGICAL فقط.

٨ ـ ٣ ـ ٢ السجلات RECORDS :

وهي التي تؤلف في مجموعها الملف ، وتتكون من مجموعة من الحقول FIELDS ويمكن لأكثر من حقل أن يجتمع ليكون مجموعة ضمن السجل تسمى GROUP و يشار الى توصيف السجل في لغة كوبول برقم مستوى هو 01 وللحقول المؤلفّة له برقم مستوى محصور بين 02 و 49. ويمكن أن يشكل حقلاً واحداً أو مجموعة حقول FIELDS GROUP أو GROUP ITEMS ما يسمى بالمفتاح KEY لهذا السجل. والذي يستخدم ليعرّف السجل عن غيره من السجلات الموجودة ضمن الملف ، أو ليظهر موقع السجل ضمن الملف كما في الملفات النسبية التي ستمر معنا بعد قليل. إن التعامل مع هذا الحقل الذي يسمى المفتاح KEY يسهل جداً التعرف على السجل ومعالجته كما سنرى فيما بعد. فمثلاً يمكن أن يكون رقم الموظف الموجود في السجل التابع للملف الأساسي للرواتب هو المفتاح كما يمكن لأن يكون رقم الحساب هو المفتاح في النظام المحاسبي.

طول السجل :

يمكن تخزين السجلات ضمن الملف إما ضمن طول ثابت للسجل و يدعى عندئذ FIXED RECORD وغالباً ما يرمز له بـ F أو أن تخزن البيانات ضمن الملف في سجلات ذات أطوال متغيرة VARIABLE LENGTH و يرمز لها غالباً بـ V.

ـ السجلات ذات الطول الثابت F :

يـكون شكل وطول هذه السجلات ثابت ومتساوي ضمن الملف الواحد وموقع كل حقـل مـن حقـول الـسجلات ثابت بالنسبة لبداية كل سجل، والحجم الثابت للحقل يـكـون محسوبـاً على أسـاس أطـول احـتمال لورود هذا البيان ضمن الحقل، مثلا إذا حـددنـا حـجـم حقل الاسم في سجل ملف الرواتب بـ ٣٠ حرفاً، فهذا يعني أنه جرى دراسـة عـمـلـيـة لمـخـتـلـف احتمالات الاسماء وتم الاتفاق بين مصمم النظم والجهة المصنعة على أكبر اسم لا يرد أكثر من ٣٠ حرفاً، وهكذا بالنسبة لباقي الحقول.

إن هـذا الـنـوع مـن الـسـجـلات هـو أسهل في التصميم وأيسر في المعالجة لذا نجده الاكـثـر انـتشاراً في الاستعمال، ولكن من الضروري الاشارة الى أنه يضيّع حجم أكبر مـن الـفراغ ضمن وسط التخزين فليس من الضروري عملياً أن ترد كل الأسماء بطول ٣٠ حرفاً.

ـ السجلات ذات الطول المتغير V :

يحـتـوي على مـؤشر في أول كـل سـجـل عـادة و يـكـون بـطـول 2 BYTES و يسمى VARIABLE LENGTH INDICATOR و يـرمـز له بـ VLI، وفيه لا يشترط أن تكون كل السجلات متساو ية الأطوال ضمن الملف.

إن اسـتـخـدام عـدّة سـجلات مختلفة الأطوال ضمن البرنامج الواحد يمكن أن تجعل الـبرنامج معقداً. لذلك تستخدم السجلات ذات الطول الثابت بصورة أكثر انتشاراً عن السجلات متغير الأطوال.

وتجـدر الاشـارة الى أنـه يجـب تجـنـب البنية المعقدة للملف، مما يجعل المعالجة معقدة أيـضـاً على المبرمج وتحتاج منه لعمل تتبع وتدقيق DEBUGGING ثم تعديل لبرنامجه مما يزيد الأمر صعوبة.

وأخيراً تجدر الإشارة الى أنه يمكن لمجموعة من السجلات - سجل أو أكثر - أن تحزم لتشكل كتلة تسمى BLOCK لتسهل عملية نقل البيانات من وسط التخزين الى الذاكرة أو بالعكس أثناء المعالجة، وكذلك تزيد من فعالية أداء البرنامج بالسرعة المتزايدة مع زيادة حجم الكتلة.

وتحديد حجم الكتلة المثالي يخضع لاعتبارات خاصة بنظام التشغيل مع الحاسب وطريقة تقسيم الأقراص لاحتواء الملفات.

٨ - ٣ - ٣ الحقول FIELDS :

تقسم السجلات كما مرّ معنا سابقاً الى حقول ذات حجم معين - ثابت أو متغير حسب نوع السجل - وتحتوي هذه الحقول على أحرف CHARACTERS قد تكون ذات طبيعة رقمية و يشار اليها بالكوبول بالرمز 9. وقد تكون ابجدية و يشار اليها بالرمز A، وقد تكون مختلطة و يشار اليها بالرمز X.

٨ - ٤ تنظيم الملفات FILE ORGANIZATION

ان التنظيم الفيزيائي PHYSICAL ORGANIZATION للبيانات DATA على أوساط التخزين المختلفة (مثل الشريط أو القرص الممغنط ...) يمكن أن يعين بإيجاد علاقة بين مفتاح السجل RECORD KEY وموقع هذا السجل ضمن وسط التخزين الموجودة عليه تلك السجلات.

وهناك طرق مختلفة لترتيب السجلات داخل الملف ومن أجل كل طريقة معالجة متبعة. واختيار طريقة التنظيم تتبع غالباً الجمع بين متطلبات التحديث الناجمة (UP - TO - DATE) و بين سرعة الاستدعاء لسجلات هذا الملف.

٨ ـ ٤ ـ ١ تنظيم الملفات على أوساط تخزين متتابعة

SERIAL STORAGE MEDIUM (*)

من المفيد أولاً أن نبين الفرق بين عبارتي متتابع SERIAL ومتسلسل SEQUENTIAL.

إن الملف المتتابع ذلك الملف الذي فيه كل سجل يحتل الموضع التالي الشاغر ضمن وسط التخزين ولا يتبع أبداً أي تسلسل لأي من حقول السجل التي تعتبر مفاتيح لهذا السجل، ولا يوجد هنالك أيضاً أي علاقة بين قيمة الحقل ضمن الملف و بين موقعه الحقيقي PHYSICAL POSITION على وسط التخزين. وفي الملفات المتتابعة تعالج السجلات حسب تسلسلها الموجود أصلاً على وسط التخزين.

إن هذا التنظيم المتتابع يغاير التنظيم المتسلسل SEQUENTIAL والذي نجد فيه أن السجلات تنظم وتعالج حسب ترتيب لمفتاح ضمن السجل مصمم سابقاً.

إن كلا العبارتين «المتتابع» و «المتسلسل» تصبحان مترادفتين عند استخدام الشريط الممغنط MAGNETIC TAPE كوسط تخزين، فلأنه وسط تخزين متتابع فبامكانه حفظ السجلات متتابعة ومتسلسلة. إن الشريط الورقي والبطاقات المثقبة ـ الشريط الممغنط هي أوساط تخزين متتابعة، بينما يمكن اعتبار القرص الممغنط والاسطوانة الممغنطة MAGNETIC DRUM أوساط تخزين مباشرة DIRECT ACCESS MEDIUM وتسمى عادة DASD وهي اختصار لـ

DIRECT ACCESS STORAGE DEVICES

و بصورة أساسية يمكن تنظيم البيانات على أوساط التخزين المتتابعة باحدى الطريقتين :

- عشوائية : وفيها لا تكون السجلات ضمن وسط التخزين مرتبة وفق أي نظام أو مفتاح ـ مفهوم التتابع SERIAL ـ.

(*) N.C.C P 191

- متسلسلة SEQUENTIALY : وفيها تكون السجلات مرتبة وفق تسلسل محدد مسبقاً وفق مفتاح معين.

وكمثال على ذلك. إن ملف التعديل TRANSACTION يحوي على سجلات ترتيبها عشوائي SERIALY على وسط تخزين هو الشريط الممغنط، و بعد فرز سجلات هذا الملف - SORT - (أي ترتيبها وفق تسلسل مفتاح معين) يصبح هذا الملف ذا تنظيم تسلسلي.

إن التنظيم العشوائي للملفات المتتابعة يعتبر أبسط نموذج من أنواع تنظيم الملفات، وذلك بأن السجل الجديد المضاف للملف يتوضع ببساطة في آخر الملف، بينما الملفات المتسلسلة نجد أن سجلاتها تتبع تسلسلا معيناً حسب قيمة تصاعدية (أو تنازلية) لحقل معين ضمن السجل يعرّف بالمفتاح KEY.

ومما يجب ملاحظته أن الملفات ذات البنية المتسلسلة ORGANIZED FILES SEQUENTIALLY يمكن أن تعالج بطرق غير الطريقة المتتابعة ويمكن لهذه الملفات المتسلسلة أن تخزن متتابعة على اوساط تخزين غير متتابعة مثل القرص الممغنط.

إن الملفات الموجودة على وسط تخزين تتابعي SERIAL مثل تلك الموجودة على الشريط الممغنط تعرّف بوجود منطقة على الشريط تسمى LABEL عنوان، وهي عبارة عن معلومات مخزنة عادة في أول وآخر ملف البيانات. وكل شريط - كوحدة فيزيائية - REEL - بكرة - تحوي على عنوان لها، ويمكن للشريط الواحد أن يحتوي على أكثر من ملف MULTIPLE FILE VOLUME، أو أن ملفاً واحداً يسجل على أكثر من شريط MULIVOLUME FILE وكل شريط VOLUME يحتوي على نقطة البدء أو التحميل LOAD POINT وعلامة نهاية الشريط في النهاية END OF TAPE MARKER وهي عبارة عن قطعة شريطية من الألمنيوم موضوعة في كلا طرفي الشريط، ونقطة التحميل تشير الى بداية البيانات المسجلة على الشريط، وعلامة نهاية الشريط تدل على مكان انتهاء البيانات المسجلة على هذا الشريط.

وحدة قارئة الاشرطة

IBM TAPE DRIVE (Courtesy IBM)

File Label Number

Field

1	2	3	4	5	6	7	8	9	10	11	12	13	14
		File Identifier	File Serial Number	Volume Sequence Number	File Sequence Number	Generation Number		Creation Date	Expiration Date		Block Count	System Code	Reserved
1	2 3 4	5 … 21	22 … 27	28 … 31	32 … 35	36 … 39	40 41	42 … 47	48 … 53	54	55 … 60	61 … 73	74 … 80

Label Indentifier

Version Number of Generation

File Security

شكل شريط

(*) COBOL A PRAGMATIC APPROACH P 279

إن وجود عبارة LABEL RECORDS ARE STANDARD في الكوبول في قسم DATA DIVISION هي من أجل الملف الموجود على وسط تخزين مغناطيسي مثل الشريط في حالتنا هذه، ونجد أن الحاسب (بوساطة OS نظام التشغيل) يعمل فحص CHECK للعنوان الداخلي للشريط INTERNAL LABEL ليتأكد من أن هل الشريط قد جهز على وحدة حاملة الأشرطة أم لا؟.

٨ - ٤ - ٢ تنظيم الملفات على أوساط تخزين مباشرة - المعالجة *

ان أوساط التخزين المباشرة - المعالجة DASD تقدم حلاً وسطاً يجمع بين وسائل التخزين الرئيسية غير المحدودة و بين أوساط التخزين المتتالية، بتقديمها التسهيلات التالية :

- تخزن كمية كبيرة من البيانات على صلة مباشرة ON - LINE بالحاسب.
- تعالج و بدقة أي جزء معطى من بيانات الملف.
- تداول هذه البيانات يتم بسرعة فائقة.

ان أوساط التخزين مباشرة - المعالجة تشمل القرص المغنط Magnetic Disk والاسطوانة المغنطة Magnetic Drum البطاقة المغنطة Magnetic Card والأقراص المغنطة المرنة Floppy Disk ، وتملك هذه الاوساط مميزات لا تملكها أوساط التخزين المتتابعة، ونأخذ على سبيل المثال القرص المغنط فنجد المميزات التالية :

- إن رؤوس القراءة والكتابة يمكنها أن تمسح كل الوسط في وقت قصير جداً.
- امكانية قراءة وكتابة كتل من البيانات ** BLOCK OF DATA بحيث تعطى كل كتلة عنواناً ADDRESS خاصاً على وسط التخزين.

(*) N.C.C. P. 192

(**) BLOCK : عبارة عن مجموعة من السجلات RECORDS.

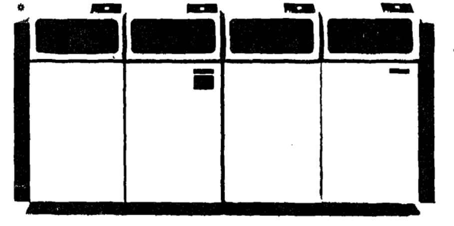
- إن السرعة الثابتة للدوران تسمح لاستغلال هذا الوسط للكتابة عليه مرات عديدة و بدقة متناهية.

وبما أن القرص الممغنط من أشهر أوساط التخزين العشوائية أو مباشرة - المعاجلة لذا سندرس البنية الفيزيائية للقرص الممغنط بصورة مختصرة.

٨ - ٤ - ٢ - ١ البنية الفيزيائية للقرص الممغنط

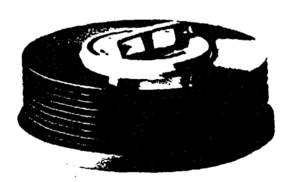
MAGNETIC DISK PHYSICAL CHARACTERISTIC

إن القرص الممغنط DISK PACK عبارة عن وسط مغناطيسي تسجل عليه المعلومات (يمكن في بعض الأنواع أن يكون القرص مؤلفاً من جزئين جزء ثابت يدعى FIXED وجزء متحرك أو متبدل يدعى REMOVABLE) و يتألف القرص من صفائح دائرية متوازية مثبتة بمحور مركزي و يركب هذا القرص على وحدة حاملة الأقراص الممغنطة تسمى DISK DRIVE والتي هي عبارة عن الوحدة التي تقرأ وتكتب من وعلى القرص الممغنط و يظهر الشكل التالي القرص الممغنط وكذلك وحدة حاملة الأقراص الممغنطة :



وحدة حاملة الأقراص الممغنطة من نوع I.B. M

(*) COBOL A PRAGMATIC APPROACH GRAUER and CRAWFORD 1978 P 269



IBM 2314 Disk Pack (*Courtesy* IBM)

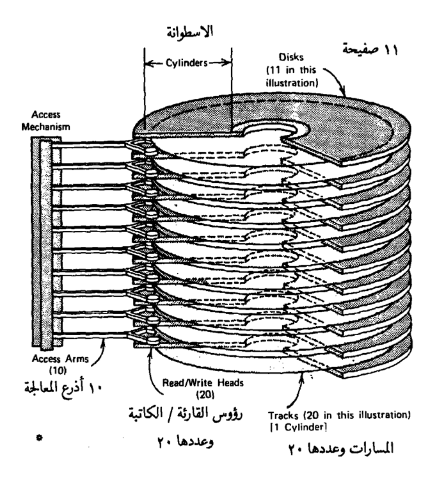
قرص ممغنط من نوع I.B.M

2311

نلاحظ من شكل القرص الممغنط أنه مؤلف من ١١ صفيحة(*) ومغطى بغطاء واقي ويمكن التسجيل على كل صفيحة أي السطح العلوي والسطح السفلي ما عدا الوجه العلوي للصفيحة الأولى ـ العلوية ـ والسطح السفلي للصفيحة ـ السفلية ـ أي يمكن استخدام عشرين سطح للتسجيل.

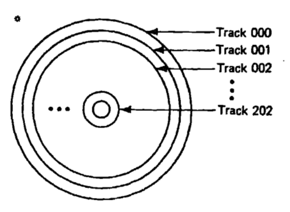
وتتم عملية الكتابة والقراءة بوساطة رؤوس قارئة ـ كاتبة تدخل بين الصفائح كما تدخل أسنان المشط بين الشعر عند تصفيفه، وهذه الرؤوس مثبتة على ذراع كما يلاحظ من الشكل التالي :

(*) هذا القرص من نوع IBM 2314 ويمكن أن توجد أقراص مختلفة الأحجام بعدد سطوح أو صفائح أكثر أو أقل وليس هذا الرقم ثابتاً.

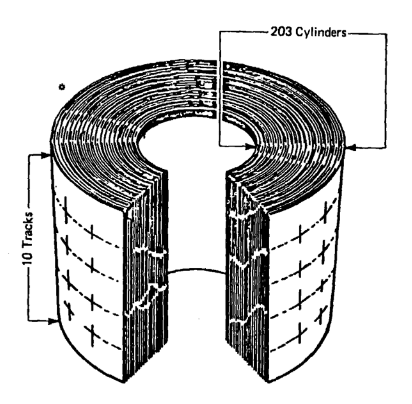


إن كل سطح تسجيل مقسم الى دوائر متحدة المركز كل دائرة تسمى مساراً TRACK والتي تسجل عليها المعلومات ولكل مسار له نفس قدرة التخزين لباقي المسارات وهكذا فان المسار الداخلي القريب من المركز له نفس قدرة التخزين للمسار البعيد عن المركز.

(٥) STERN



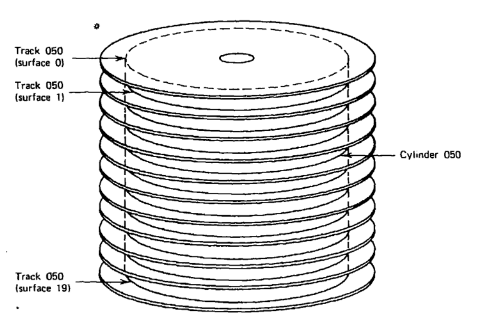
شكل مواجه لصفيحة تظهر عليها المسارات



شكل آخر يوضح المسار والاسطوانة Cylinder

(٥) ' STERN

إن مجموع مسارات أفقية معينة في القرص كله تسمى بـ CYLINDER مثال : المسار رقم ٥٠ في السطح رقم ١ والمسار رقم ٥٠ في السطح رقم ٢ والمسار رقم ٥٠ في السطح رقم ٣ ... والمسار رقم ٥٠ في السطح رقم ٩ تؤلف في مجموعها (عمودياً) مفهوم الـ CYLINDER أنظر الشكل التالي :



ويمكن أن نذكر أنه في بعض الحاسبات يقسم كل مسار TRACK الى أجزاء أصغر منه تسمى قطاعاً SECTOR ، وهذا القطاع هو الذي تسجل عليه البيانات وهو يعادل أو يقارب مفهوم كتل البيانات BLOCKS لكن هنا التقسيم من الناحية الفيزيائية.

٨ ـ ٤ ـ ٢ ـ ٢ القدرة التخزينية للأقراص الممغنطة CAPACITY **

إن الحد الأقصى لاستيعاب وحدة الاقراص الممغنطة يحدد بثلاثة عوامل :

أ ـ كثافة التسجيل (عدد الأحرف BYTES التي تسجل على المسار TRACK).

(*) STERN

(**) COBOL A PRAGMIATIC APPROACH 1978 Grauer and Crowford P 297 - 298

ب ـ عـدد أسـطـح الـتـسجـيـل RECORDING SURFACE (عـدد المسـارات بكل CYLINDER).

جـ ـ عدد المسارات على أسطح التسجيل (عدد الـ CYLINDERS).

وتزداد قدرة التخزين بزيادة أي عامل من العوامل الثلاثة السابقة.

٨ ـ ٤ ـ ٢ ـ ٣ مفهوم المساحة الفائضة OVERFLOW AREA

إن المنحنى الطبيعي للملفات الأساسية MASTER FILES هي التوسع في الحجم (expand). ويمكن أن يتم التوسع أيضاً بزيادة عدد الملفات المخزنة على القرص. أو بزيادة حجم هذه الملفات، وفي كلا الحالتين يمكن أن يصل القرص الى حد الاشباع أو حالة الفيضان OVERFLOW، لذا نجد أنه في كل CYLINDER أو كل TRACK تخصص مساحة تسمى OVERFLOW AREA يمكن أن تساعد على حل مؤقت لمشكلة التوسع، فالملفات عند انشائها CREATION يُحدد لها مساحة معنية، فاذا وصل هذا الملف لحد الاشباع أي استهلكت المساحة المعطاة له (أو في بعض حالات التنظيم العشوائي أن أكثر من مفتاح يكون له نفس عنوان التخزين) لذا يتم تسجيل السجل الجديد في المساحة الاضافية OVERFLOW AREA مع تعريف الملف بعنوانه في تلك المساحة.

وفي أحيان كثيرة يكون «الفيضان» للمعلومات بمكان معين في القرص و باقي الأماكن تكون شاغرة. هذه الحالة تنشأ من أنه يمكن لبعض الملفات أن تكون نشطة ـ أي يجري عليها عمليات TRANSACTIONS بصورة دورية فتمتلىء المساحة المخصصة للملف وكذلك المساحة الاضافية في حين تكون بعض الملفات خاملة أي أن العمليات التي تجري عليها قليلة، أو أنها تكون ذات صفة مصدرية قليلة التعديل، لذا فتكون مساحتها الاضافية فارغة، وجزء من المساحة الأصلية أيضاً.

لذا نجد أن أحد الحلول المطروحة للاستغلال الأمثل للفراغ SPACE الموجود على القرص هي اعادة تنظيم القرص REORGANIZATION . أو على الأقل إعادة تنظيم بعض الملفات الموجودة ضمنه بإعطاء الملفات النشطة حجماً أكبر والملفات الخاملة حجماً أقل.

وتجدر الإشارة أنه عند إعادة تنظيم القرص يستعان بوسط تخزيني آخر وسيط قد يكون قرصاً آخراً أو شريطاً معناطيسياً.

الآن و بعد أن عرفنا البنية الفيزيائية للقرص سندرس بشيء من التفصيل تنظيم الملفات على هذا الوسط.

يمكن أن تنظم الملفات على وسط التنظيم مباشرة ـ المعالجة بأحد أربعة طرق :

أ ـ تنظيم متسلسل STANDARD SEQUENTIAL ORGANIZATION

بـ ـ تنظم مفهرس INDEXED ORGANIZATION

جـ ـ تنظيم مباشر DIRECT ORGANIZATION

د ـ تنظيم نسبي RELATIVE ORGANIZATION

أ ـ التنظيم المتسلسل : هو مشابه تماماً للتنظيم الموجود على أوساط تخزينية متتابعة مثل الشريط، فكل سجل يأتي مباشرة بعد السجل الذي يسبقه إما بترتيب معين حسب مفتاح ـ متسلسل ـ أو بشكل عشوائي ـ متتابع ـ وتتم معالجته تماماً كما لو أنه على شريط ممغنط، وله عنوان بداية ونهاية أيضاً ـ LABEL ـ. إن هذا التنظيم يستغل المساحة المعطاة للملف استغلالا أمثل ولا يترك فراغات بين السجلات والتحديث عليها يتم مثل التحديث على نفس الملفات كما لو كانت على شريط ممغنط.

ب ـ تنظيم الملفات المتسلسلة المفهرسة **Indexed sequential files organization**(*)

إن الملفات المتسلسلة المفهرسة هي الطريقة في التنظيم التي تسمح للمبرمج معالجة هذا النوع من الملفات والموجود على القرص الممغنط بشكل عشوائي RANDOM. وتعتمد هذه الطريقة في التنظيم على ايجاد ملف جزئي يلحق بالملف الأصلي ـ أي ملف البيانات، DATA FILE ـ هذا الملف الجزئي يحتوي على مفاتيح سجلات الملف RECORD KEYS مرتبة بشكل متسلسل.

DATA FILE

	DATA	KEY
ADDRESS 3		10
ADDRESS 5		1
ADDRESS 8		2

INDEX FILE

KEY	ADDRESS
1	5
2	8
10	3
12	
22	
23	
27	
35	
80	
82	

(*) تعرّف احياناً في بعض الحاسبات بالرمز ISAM وفي I.B.M تحت نظام تشغيل OS/MVS تدعى VSAM.

ويحتوي بالإضافة الى ذلك عنوان سجل كل مفتاح على القرص.

ان كلاً من الملف الجزئي ـ الفهرس ـ وملف البيانات يؤلفان ملفاً واحداً يسمى Indexed sequential file وما هذا التقسيم إلاّ ضمني، أما ما يخص مستخدم الحاسب USER أو المبرمج Programer في التوصيف وانشاء الملفات وغيرها فهو يتعامل مع اسم واحد فقط وهو اسم الملف المفهرس في البرنامج.

ولتتضح لنا الصورة بشكل أكبر نضرب المثل التالي:

إذا أردنا البحث عن موضوع معين ضمن كتاب؛ فإما أن نقرأ الكتاب صفحة صفحة حتى نصل للبحث المطلوب. أو ننظر في فهرس الكتاب Index ونأخذ منه رقم الصفحة ـ العنوان ـ ونفتح مباشرة على البحث المطلوب.

ولا ننسى أن كلاً من المعلومات في الكتاب والفهرس هما تحت عنوان واحد.

أخيراً نشير الى أنه لكل سجل لا بد أن يكون هناك مفتاح ولهذا المفتاح قيمة وحيدة ضمن الملف UNIQUE و يسمى PRIME KEY المفتاح الأولي أو الأساسي؛ ويمكن للسجل أن يعرّف بمفاتيح ثانوية وثالثية حسب امكانيات كل حاسب ويمكن لهذه المفاتيح ـ الثانوية ALTERNATE KEYS أن تكون قيمتها مكررة DUPLICATES في الملف.

جـ تنظيم الملفات المباشرة DIRECT FILE ORGANIZATION*

يعتمد هذا الأسلوب من التنظيم على أجراء بعض العمليات الحسابية RANDOMIZING على الحقل الذي يسمى مفتاح سجل RECORD KEY ومن ناتج

(*) هذا النوع من التنظيم غير شائع في كل الحاسبات و يستعاض عنه بالتنظيم المفهرس المتسلسل INDEXED SEQUENTIAL، و يستخدم هذا النوع من التنظيم في أنظمة قواعد البيانات DATA BASE بصورة أساسية.

هـذه العـملية الحسابية يعرف عنوان السجل على القرص الممغنط حسب رقم الاسطوانة CYLINDER والمسـار TRACK ويخزن في ذلك العنوان. إن الطريقة هذه لا تعتمد على قراءة متسلسلة من ملف البيانات أو من ملف المفهرس وإنما على هذه العملية الحسابية لذلك سميت DIRECT لأنه مباشرة ينفذ للعنوان المطلوب.

وتجدر الإشارة أن هـذه العـمليات الحسابية RANDOMIZING ليست بالصيغة البسيطة التي نتصورها، إنما تعتمد أساساً على ايجاد صيغة رياضية معقدة من أجل ايجاد عـنوان وحيد UNIQUE لكل سجل جديد مضاف الى الملف. وكثيراً من الأحيان نجد أن الوقـت اللازم لانجـاز هذه العملية أكبر من الوقت المصروف على البحث في ملف مـفـهـرس، لـذلـك نـجـد في أغـلـب الـتطبيـقات أن الملفات المفهرسة Indexed sequential files أكثر استعمالا من Direct files.

د ـ تنظيم الملفات النسبية RELATIVE FILE ORGANIZATION

في هـذا التـنـظـيم نجد أن كل سجل معرف برقم ذي علاقة نسبية مع بداية الملف يسـمـى RELATIVE RECORD وهـو وحيد ضمن الملف. إن هذا التنظيم مسموح به فـقـط على الـقـرص الممغنط. إن رقم السجل رقم صحيح INTEGER واكبر من الصفر والسجلات تخزن وتستدعى اعتماداً على هذا الرقم وتبدأ من الرقم ١.

فـالـسجـل رقـم ١٠ هـو ذاك السجل ذو الترتيب العاشر في الملف؛ والسجل الذي رقـمـه ٢١٥ هـو ذاك السجل الذي ترتيبه مئتان وخمسة عشر في الملف والسجل رقم ٢ هو السجل ذو الترتيب الثاني في الملف ... وهكذا.

٨ ـ ٥ ـ أساليب معالجة الملفات FILE ACCESS METHODS

تختلف أساليب معالجة الملفات باختلاف أوساط التخزين، فكما مرّ معنا في الجزء السابق (٤) نجد أن هنالك نوعين من أوساط التخزين:

أ ـ أوساط تخزين متتابعة SERIAL STORAGE DEVICES

ب ـ أوساط تخزين مباشرة المعالجة DIRECT ACCESS STORAGE DEVICES

٨ ـ ٥ ـ ١ معالجة الملفات المخزنة على أوساط تخزين متتابعة

ان الملفات المتسلسلة التنظيم SEQUENTIAL ORGANIZATION . المخزنة على أوساط تخزين متتابعة مثل الشريط الممغنط أو البطاقات المثقبة تعالج بصورة بسيطة جداً ، إذ تتم قراءة كل سجل وتجرى عليه العمليات المطلوبة . نفس الشيء ينطبق على الملفات المتتابعة Serial files .

ومن أجل توصيفها ومعالجتها يرجع للفصول السابقة* من أجل شرح هذه التعليمات أو التوصيف .

ونذكر بصورة مختصرة وسريعة التعليمات الأساسية التي تعالج هذه الملفات :

١ ـ تعليمة فتح الملف

OPEN { INPUT / OUTPUT / EXTEND } file name 1,....

٢ ـ تعليمة القراءة

READ file - name [INTO identifier] AT END
imperative statements

٣ ـ تعلمية الكتابة

WRITE Record - name [FROM identifier]
[AFTER ADVANCING n LINES]

٤ ـ تعليمة اغلاق الملف

CLOSE file name 1, ...

٥ ـ ونذكر بتوصيف الملف في INPUT - OUTPUT SECTION.

(*) الفصل الثاني من القسم الأول

```
FILE - CONTROL.
        SELECT file - name
        ASSIGN TO implementor - name
        ORGANIZATION IS SEQUENTIAL,
        ACCESS MODE IS SEQUENTIAL.
```

ORGANIZATION IS SEQUENTIAL,	نوع تنظيم الملف تسلسلي
ACCESS MODE IS SEQUENTIAL.	اسلوب المعالجة تسلسلي

فعند طباعة ملف مثلاً، يقرأ الملف وتحرك الحقول منه الى حقول الطباعة وتتم الكتابة على الطابعة باستخدام تعليمات :

```
OPEN
READ ....
MOVE ....
WRITE ...
```

وهذا أبسط شكل معالجة؛ لكن عندما نريد الاستفسار عن سجل محدد عندها يجب أن نقرأ الملف سجلاً سجلاً حتى نحصل على السجل المطلوب فلو كان عندنا ١٠٠٠ الف سجل في ملف الرواتب وأردنا الاستفسار، أو طباعة الموظف رقم ٩٠٠ الموجود سجله في الملف؛ فيجب علينا أن نقرأ ٨٩٩ سجلاً قبله حتى نصل اليه ثم نطبع السجل أو نعالجه حسب المطلوب؛ إن المشكلة تزداد صعوبة إذا طلب منا موظف آخر ذو رقم ٢٠٠ فالتعليمات المتوفرة واسلوب تنظيم الملف ومعالجته لا تسمح بالقراءة للخلف أي كل قراءة تأخذ السجل الذي يلي السجل الحالي، فالحل الوحيد لمعالجة الموظف رقم ٢٠٠ بعد أن عالجنا الموظف رقم ٩٠٠ هو إغلاق هذا الملف بتعليمة CLOSE ثم إعادة فتحه بتعليمة OPEN ثم القراءة ١٩٩ مرة حتى نصل للسجل رقم ٢٠٠ المطلوب (يتم اغلاق وفتح الملف حتى نحصل على السجل الأ ول من الملف).

٨ ـ ٥ ـ ١ ـ ٢ تحديث الملفات المتسلسلة UPDATE SEQUENTIAL FILES

يقصد بعملية التحديث هو تعديل محتوى الملف بما يتناسب مع آخر معلومات صحيحة عن الملف.

فلو أخذنا ملف البيانات الشخصية في نظام الشئون الشخصية PERSONNEL نجد أنه يحتوي على سجلات الموظفين، كل سجل يحتوي على حقول تخص حالة الموظف، اسمه، رقمه، عنوانه، حالته الاجتماعية، .... الى ما هنالك من معلومات.

إن الملف بحد ذاته عرضة للتغير كأن يعين موظف جديد، إذاً سنضيف سجلاً الى الملف؛ أو قد يترك العمل موظفاً آخر، أي سنحذف سجلا من الملف، وقد تكون المعلومات نفسها بحاجة لتغيير، لو فرضنا أن موظفاً غير عنوانه، أو تزوج أو طلّق، فالحقول الموجودة ضمن الملف تحتاج الى تحديث.

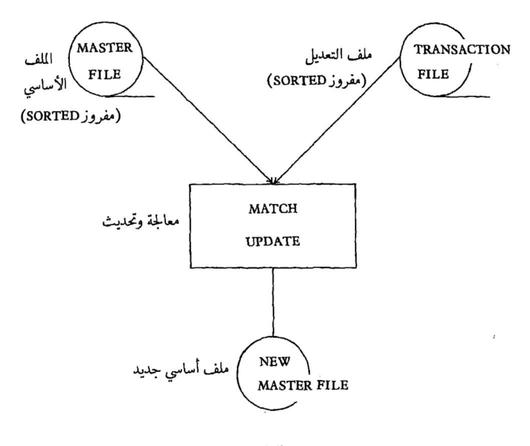
إن الامكانيات المتاحة في الملفات المتسلسلة والموجودة على أوساط تخزين متتابعة مثل الشريط الممغنط لا تسمح بتبديل محتوى الملف، أو حذف سجل مباشرة منه أو اضافة سجل مباشرة اليه.

إذاً يجب علينا الاستعانة بملفين آخرين، ملف فيه الوقوعات أو التبديلات TRANSACTIONS و يسمى ملف التعديل و يوجد فيه رقم السجل ـ الموظف ـ المراد معالجته، ونوع العملية المرغوب فيها وتسمى غالباً TRANSACTION CODE وهي حقل ضمن سجل ملف التعديل؛ وملف آخر هو ملف وليد عملية دمج الملفين السابقين ملف الشئون الشخصية مع ملف التعديل، وهذا الملف الوليد يحتوي على السجلات التي لم يطرأ عليها أي تعديل الموجودة في ملف الشئون الشخصية بالاضافة الى التعديلات المرغوب فيها والقادمة من ملف التعديلات مثل إضافة سجلات جديدة، أو سجل تعديل لسجل أساسي.

إن الملف الوليد هذا هو عبارة عن ملف اساسي جديد NEW MASTER FILE لذا يجري إلغاء الملف الأساسي القديم ـ أو الاحتفاظ به بعيداً عن المعالجة الحالية ـ وتعاد تسمية الملف الوليد بالملف الأساسي و يصبح هذا الملف هو الملف الأساسي MASTER FILE و يكون محدّثاً UP - TO - DATE .

وتجدر الاشارة أنه يجب قبل البدء في معالجة الملفات السابقة ـ الأصلي القديم وملف التعديل ـ أن يكونا مفروزين SORTED على مفتاح معين في كلا الملفين في حالتنا هنا على رقم الموظف، وعلى نفس الاتجاه (تصاعدي أو تنازلي).

أنظر البرنامج التالي والذي فيه يتم تحديث ملف أساسي من ملف حركة.



```
*
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT22.
 AUTHOR.        KHAYAT-JNAID.
*
* يقوم هذا البرنامج بقراءة ملف اساسي وملف
* تعديل ثم ينشيء ملف اساسي جديد
*
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.    IBM-370.
   OBJECT-COMPUTER.    IBM-370.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
       SELECT MASTRF  ASSIGN  TO MSTF
       ORGANIZATION   SEQUENTIAL;
       ACCESS MODE    SEQUENTIAL.
       SELECT TRANSF  ASSIGN  TO TRNF
       ORGANIZATION   SEQUENTIAL;
       ACCESS MODE    SEQUENTIAL.
       SELECT NMASTRF  ASSIGN  TO NMSTF
       ORGANIZATION   SEQUENTIAL;
       ACCESS MODE    SEQUENTIAL.
 DATA DIVISION.
 FILE  SECTION.
* توصيف الملف الاساسي
 FD     MASTRF     LABEL RECORD  IS STANDARD
                       DATA RECORD IS M-REC.
 01     M-REC.
        03     M-NO        PIC  X(5).
        03     M-DATA      PIC  X(35).
* توصيف ملف التعديل
 FD     TRANSF     LABEL RECORD  IS STANDARD
                       DATA RECORD IS T-REC.
 01     T-REC.
        03     T-NO        PIC  X(5).
        03     T-DATA      PIC  X(34).
* يعتبر حقل T-CODE هو حقل رمز العملية
        03     T-CODE      PIC  X.
* توصيف الملف الاساسي الجديد
 FD     NMASTRF     LABEL RECORD  IS STANDARD
                       DATA RECORD IS N-REC.
```

```
 01      N-REC.
         03     N-NO        PIC  X(5).
         03     N-DATA      PIC  X(35).
*
 PROCEDURE DIVISION.
        OPEN          INPUT  MASTRF TRANSF
                      OUTPUT        NMASTRF.
*       القراءة من الملفين
*
        PERFORM READ-MSTR THRU EXIT1.
        PERFORM COMPAR THRU EXIT-LOOP
                UNTIL   T-NO = M-NO
                AND     T-NO = HIGH-VALUE.
        CLOSE MASTRF
                    TRANSF
                       NMASTRF.
        STOP    RUN.
*


*
*         مقارنة قيمة المفتاح في السجلين
*  وكتابة السجل الذي مفتاحه اصغر علي
*  ملف الإخراج ثم القراءة منه من جديد.
*  اما في حالة المساواة فاذا كان رمز
*           العملية D يعني حذف فيقرأ من
*(الملفين من جديد (اي نتجاوز الكتابة
*اما حالة التعديل C فينقل محتوى السجل
*  من ملف التعديل الى الاساسي الجديد.
*
 COMPAR.
       IF          M-NO LESS THAN T-NO
                   WRITE N-REC FROM M-REC
                   PERFORM      READ-MSTR
          ELSE
              IF T-NO LESS THAN M-NO AND
                 T-CODE = 'A'
                 PERFORM MOVE-2 THRU EXIT2
        ELSE
              IF T-NO =   'D'
                   PERFORM READ-TRNS THRU EXIT1
```

```
        ELSE
            IF T-NO =  'C'
                PERFORM MOVE-2 THRU EXIT1.
 EXIT-LOOP.
            EXIT.
*
 MOVE-2.
      MOVE  T-NO        TO N-NO.
      MOVE  T-DATA      TO N-DATA.
      WRITE N-REC.
*        القراءة من ملف التعديل
 READ-TRNS.
      READ  TRANSF  AT END
            MOVE   HIGH-VALUE TO T-NO.
 EXIT2.
      EXIT.
*        القراءة من الملف الاساسي   *
 READ-MSTR.
      READ  MASTRF  AT END
            MOVE   HIGH-VALUE TO M-NO.
 EXIT1.
      EXIT.
*
```

إن الفكرة الأساسية في هذا البرنامج، برنامج التحديث هي القراءة من الملف الأساسي ومن ملف التعديل ومقارنة المفتاحين ونكون أمام الاحتمالات التالية :

أ ـ فاذا كان مفتاح الملف الأساسي أصغر من مفتاح ملف التعديل فهذا يعني أن هذا السجل الأساسي ليس عليه أي تعديل فينقل الى الملف الجديد ـ الوليد ـ و يكتب عليه و يرجع للقراءة من الملف الأساسي ثانية.

ب ـ اذا كان مفتاح ملف التعديل أصغر من مفتاح الملف الأساسي فهذا يعني أن ملف التعديل يحمل سجلاً لا يوجد مثيل له في الملف الأساسي أي سجلاً جديداً فينقل هذا السجل من ملف التعديل الى الملف الجديد الوليد و يكتب و يقرأ من ملف التعديل ثانية، على شرط أن يكون رمز العملية 'A'.

جـ ـ في حال كون المفتاحين متساويين فهذا يعني أن للسجل الأساسي سجل تعديل وهذا التعديل يكون أحد أمرين :

جـ ـ ١ أما أن رمز العملية TRANSACTION CODE في سجل التعديل يحتوي على رمز الحذف (و بالفرض) الحرف D عندها نعاود قراءة الملفين من جديد ونكون بذلك قد مررنا على السجل الموجود في الملف الأساسي دون نقله للملف الوليد ـ BY PASS ـ أي حذفناه .

جـ ـ ٢ أو أن رمز العملية TRANSACTION CODE في سجل التعديل يحتوي على رمز التغيير C مثلاً ، عندها يكون سجل ملف التعديل يحتوي على معلومات جديدة عن نفس السجل الموجود في الملف الأساسي ، لذلك ننقل المعلومات من سجل التعديل الى الملف الوليد ـ الجديد ـ ونعاود قراءة الملفين معاً* .

قد تصادف بعض المبرمجين أو الطلاب مشكلة نهاية الملفات END OF FILES فلو انتهى ملف التعديل أولاً فيجب نقل المتبقي من الملف الأساسي الى الملف الجديد وكتابته (أي سجلات ليس عليها تعديلات) أما لو انتهى الملف الأساسي أولاً فهذا يعني أن المتبقي من ملف التعديل هو سجلات جديدة للاضافة لذلك تنقل الى الملف الجديد ـ الوليد ـ وتكتب ، إن معالجة ذلك في برنامج كوبول قد يسبب إرباكاً عند غلق الملف المنتهي والبدء بروتين جديد لمعالجة المتبقي من أحد الملفين . لكن اذا استعملنا القيمة العليا HIGH-VALUE** للمقارنة (بفرض أن الملفين مفروزان تصاعدياً) عند انتهاء أحد الملفين فيكون الملف الآخر غير المنتهي حالة أصغر بالنسبة للملف المنتهي

(*) قد تنص المسألة إلى أنه يوجد أكثر من سجل تعديل للسجل الأصلي عندها نعاود القراءة من ملف التعديل فقط ، وهذا الأسلوب غالباً ما يكون في النظم المحاسبية .

(**) قيمة معرفة في الحاسب ولا داعي لوصفها في WORKING-STORAGE وتماثل في تعريفها كلمة SPACE أو ZERO مع اختلاف محتواها طبعا ، وهي قيمة ثابتة في لغة كوبول تعادل أعلى قيمة يمكن وصفها في الذاكرة وتسمى FIGURATIVE CONSTANT .

والذي يحوي على القيمة العليا HIGH - VALUE فنعود للحالة أ أو ب ولا حاجة لكتابة تعليمات جديدة ـ أنظر البرنامج ـ وعند تساوي مفتاحي الملفين مع HIGH - VALUE يكون قد انتهى الملفان فنقفلهما وننهي العمل .

يمكن أن نشير الى أنه قد يطلب عمليات مختلفة على برنامج تحديث الملف الأساسي كالكشف عن التكرار في ملف التعديل وغيرها من طلبات تعالج عندها حسب المطلوب .

إن اسلوب المعالجة المتسلسلة هو الأسلوب الوحيد المتوفر في معالجة الملفات الموجودة على أوساط تخزين متتابعة مثل الشريط الممغنط والبطاقات المثقبة ، بينما نجد أن هنالك أكثر من طريقة أو أسلوب معالجة على الملفات الموجودة على أوساط تخزين مباشرة ـ المعالجة DASD ، كما سنرى في الفصل التالي .

وأخيراً تجدر الاشارة الى أن هذا النوع من المعالجة يسمى نمط المعالجة غير المباشرة BATCH MODE أو OFF - LINE والتي من مساوئها البطيء في استدعاء سجل معين وتعقيد العمليات التي يجديها المشغل بنقل ملفات من وإلى الحاسب وعدم امكانية التعديل على نفس الملف .

٨ ـ ٥ ـ ٢ معالجة الملفات المخزنة على أوساط تخزين مباشرة ـ المعالجة DASD

تقسم أنواع المعالجة التي يمكن أن تجري على الملفات الى قسمين أساسيين :

١ ـ المعالجة المتسلسلة SEQUENTIAL ACCESS

٢ ـ المعالجة العشوائية (المباشرة) RANDOM ACCESS (DIRECT)

ويمكن أن نعالج ملفاً معيناً بالطريقتين معاً و بنفس البرنامج و يسمى عندئذ اسلوب المعالجة بـ المعالجة الحركية :

٣ ـ المعالجة الحركية DYNAMIC ACCESS

٨ ـ ٥ ـ ٢ ـ ١ المعالجة المتسلسلة :

وفيها تتم معالجة سجلات الملف بصورة متسلسلة وكل سجل يقرأ ثم يعالج، وتطبق على الملفات ذات التنظيم التسلسلي، وذات التنظيم المفهرس، وذات التنظيم المباشر DIRECT، وكذلك الملفات ذات التنظيم النسبي.

٨ ـ ٥ ـ ٢ ـ ١ ـ ١ معالجة الملفات المتسلسلة SEQUENTIAL FILES

إن اسلوب وتوصيف المعالجة المتسلسلة للملفات المتسلسلة هو نفس الاسلوب المتبع في المعالجة المتسلسلة للملفات الموجودة على وسط تخزين متتابع والموصوفة في الفقرة ٨ ـ ٥ ـ ١ واسلوب التحديث عليها هو نفسه الموصوف في الفقرة ٨ ـ ٥ ـ ١ ـ ٢، لذلك راجع الفقرتين المذكورتين من أجل معرفة معالجة الملفات المتسلسلة تسلسلياً.

٨ ـ ٥ ـ ٢ ـ ١ ـ ٢ معالجة الملفات المفهرسة INDEX SEQUENTIAL FILES

إن اسلوب المعالجة للملفات المتسلسلة والمفهرسة هو قريب من المعالجة المتسلسلة للملفات المتسلسلة مع بعض الاختلافات التي سنذكرها هنا :

ـ في قسم IDENTIFICATION DIVISION لا نلاحظ أي اختلاف بالتوصيف.

ـ في قسم ENVIRONMENT DIVISION نلاحظ أنه في FILE - CONTROL أصبح التوصيف كالتالي :

```
SELECT file - name ASSIGN TO implementor - name,
ORGANIZATION IS INDEXED,
ACCESS MODE IS SEQUENTIAL
RECORD KEY IS data - name - 1 *
```

(*) بعض الحواسب مثل TI تشترط بتوصيف المفتاح الأولي أن يكون من نوع ALPHANUMERIC.

[ALTERNATE RECORD KEY IS data - name - 2]

[WITH DUPLICATES] [FILE STATUS IS data - name - 3].

نلاحظ من التوصيف السابق :

أن نوع تنظيم الملف هو مفهرس ORGANIZATION IS INDEXED واسلوب المعالجة هو متسلسل ACCESS MODE IS SEQUENTIAL وبما أنه مفهرس إذاً يجب أن يكون له مفتاح KEY و بالتالي نضع اسم المفتاح مكان data - name - 1 ـ نذكر أن المفتاح هو حقل في السجل ضمن الملف ـ هذا المفتاح طبعاً معرّف في قسم الملفات DATA DIVISION* ضمن توصيف السجل. و data - name - 1 تدل إذاً على المفتاح الأساسي الأولي Prime key للملف.

ويمكن أن تكون هنالك مفاتيح ثانوية فتوصف عندئذ في عبارة :

ALTERNATE RECORD KEY IS data - name - 2,

وإذا لم يوجد للسجل سوى مفتاح أساسي واحد فلا داعي لتوصيف المفتاح الملحق. أما عبارة WITH DUPLICATES فتفيد بتعريف الحاسب أن هذا المفتاح الثانوي ـ وليس الأساسي قطعاً ـ ليس وحيداً بالملف بل مكرراً DUPLICATE. فاذا لم توضع هذه العبارة اعتبر المفتاح الثانوي وحيداً أيضاً UNIQUE، فاذا صادف أن وصفنا المفتاح الثانوي بأنه وحيد وأردنا اضافة سجل للملف له نفس قيمة المفتاح الثانوي حصلنا على حالة خطأ.

يمكن تعريف أكثر من مفتاح ثانوي لنفس الملف؛ إنما العدد يختلف من حاسب لآخر فنجده في TI مثلاً بحدود ١٣ مفتاح ثانوي.

- في قسم البيانات DATA DIVISION لا يوجد أي توصيف خاص.
- في قسم [illegible] PROCEDURE DIVISION

(*) بعض الحواسب [illegible] تشترط بتوصيف المفتاح الأولي أن يكون من نوع ALPHANUMERIC.

ولنستعرض التعليمات الأساسية التي يمكن أن يحصل فيها بعض التغيير :

أ ـ تعليمة فتح الملف OPEN

```
            ┌ INPUT  ┐
*OPEN       │ OUTPUT │      file name 1, ....
            └ I - O  ┘
```

نلاحظ أنه يمكن فتح الملف المفهرس إما بشكل INPUT و بالتالي نستطيع أن نقرأ منه ولا نستطيع أن نكتب عليه ؛ أو بشكل OUTPUT أي نستطيع أن نكتب عليه دون أن نقرأ منه ؛ والشكل الثالث وهو جديد I - O أي INPUT - OUTPUT أي يمكن أن نقرأ منه ونكتب عليه بنفس البرنامج وهذه ميزة الملفات المفهرسة . بعد تنفيذ هذه التعليمة بنجاح يكون المؤشر POINTER الخاص بالملف متوضعاً عند بداية أول سجل في الملف .

ونذكر هنا أنه من جملة الأخطاء التي يواجهها المبرمج عند تنفيذ برنامج يحوي على سجل مفهرس أن تعليمة OPEN لا تنفذ بنجاح و يكون على الغالب توصيف الملف في عبارة SELECT أو في عبارة FILE DESCRIPTION مغايراً للمعلومات التي اعطيت للحاسب عند انشاء الملف فيزيائياً على القرص . كأن يكون طول المفتاح مغايراً للطول الموصف في البرنامج ، أو أن يكون هناك تكرار في المفتاح الثانوي ولم يذكر ذلك المبرمج في عبارة ALERNATE KEY WITH DUPICATES . لذلك يجب الانتباه لتوصيف الملف ومفاتيحه في البرنامج بأن تكون مطابقة تماماً للمعلومات التي أعطيت للحاسب عند انشاء هذا الملف فيزيائياً على الوسط .

(*) TI990 COBOL MANUALS

ب ـ تعليمة القراءة READ

نجد أن الشكل للتعليمة

```
*READ file name   NEXT   RECORD [INTO identifier]
        [AT END  imperative - statement.]
```

هذه التعليمة تطبق عندما يراد القراءة من ملف متسلسل ومفهرس بطريقة متسلسلة ACCESS MODE SEQUENTIAL.

نلاحظ هنا أن الفرق بين هذه التعليمة وتعليمة READ المستخدمة في المعالجة المتسلسلة للملفات المتسلسلة ـ غير المفهرسة ـ هو وجود عبارة NEXT أي السجل التالي.

هذه التعليمة تنفذ اذا كان الملف مفتوح بشكل INPUT أو I - O فقط.

جـ ـ تعليمة الكتابة WRITE

الشكل العام للتعليمة :

```
*WRITE  record - name [FROM identifier]
     INVALID KEY  imperative - statement
```

تستخدم هذه التعليمة للكتابة على ملف مفهرس ذي اسلوب معالجة متسلسلة أو عشوائية أو حركية. و يشترط في حالتنا هذه اي المعالجة المتسلسلة للملفات المفهرسة بأن تكون قيم مفتاح السجل أعلى من قيمة مفتاح السجل الموجود قبله أي لو كان آخر سجل موجود في الملف قيمة مفتاحه ١٢٥ فيجب أن تكون قيمة المفتاح للسجل المضاف أعلى من ١٢٥ وإلا فتتشكل حالة INVALID KEY كما يجب أن يكون الملف مفتوحاً بشكل OUTPUT أو I - O.

(*) TI990 COBOL MANUALS

مثال :

```
WRITE OUT - REC INVALID KEY
    DISPLAY "RECORD NOT ADDED" GO TO ERRRTN.
DISPLAY "SUCCESSFUL INSERTION".
MOVE ...
```

من المثال نلاحظ أنه اذا تمت الاضافة بنجاح يظهر المبرمج مثلاً رسالة تشعر بذلك و يتابع المعالجة؛ وإذا لم تتم وتشكل خطأ نجد أن المبرمج أظهر خطأ وذهب الى فقرة ببرنامجه اسمها ERR - RTN ، والذهاب لهذه الفقرة لا يكون إلا عندما تتشكل حالة الخطأ.

د. تعليمة اعادة الكتابة REWRITE*

تستخدم هذه التعليمة لتعديل محتوى سجل معين ضمن الملف المفهرس. ففي الملفات ذات التنظيم التسلسلي نجد أنه لا نستطيع الكتابة عليها ، أو اعادة الكتابة إنما نستخدم ملفاً ثالثاً ليكون هو حاصل التعديل - انظر الفقرة ٨ - ٥ - ١ والمثال - بينما هنا في الملفات المفهرسة نستطيع أن نضيف سجلات جديدة كما نستطيع أن نبدل محتوى سجل معين ونعيد كتابة السجل القديم بمحتواه الجديد. أما التعديل فقد يأتي من سجل آخر أو عن طريق الشاشات.

والشكل العام للتعليمة :

REWRITE record - name [FROM identifier].

يشترط لتنفيذ هذه التعليمة أن يكون الملف مفتوحاً بشكل I - O .

(*) TI990 COBOL MANUALS

مثال :

REWRITE IN - REC.

هـ - تعليمة البداية START*

هذه التعليمة توفر للمبرمج الأساس الذي سيبدأ منه المعالجة المتسلسلة للسجلات ضمن الملف.

فلو أراد المبرمج أن يبدأ المعالجة من قيمة معينة للمفتاح وليس من أول الملف فان استخدام تعليمة START تحدد موقع بدء المعالجة.

الشكل العام لها :

```
                          ┌          ┌ EQUAL TO         ┐             ┐
                          │          │    =             │             │
                          │          │ GREATER THAN     │             │
START file - name         │ KEY IS  <  >                 > data - name │
                          │          │ NOT LESS THAN    │             │
                          └          └ NOT <            ┘             ┘

           [INVALID KEY imperative statement.]
```

حيث أن file - name هو اسم الملف الذي تجري فيه المعالجة ؛ ويجب أن تكون المعالجة SEQUENTIAL أو DYNAMIC ويجب أن يكون مفتوحاً بشكل INPUT أو I - O.

بعد تنفيذ هذه التعليمة وإذا لم يتحقق الشرط المطلوب نكون أمام حالة INVALID KEY.

(*) TI990 COBOL MANUALS

مثال :

```
START VALFIL KEY IS > REC - KEY    INVALID
        KEY GO TO WRONG - RTN.
```

أما data - name فهي تشير الى حقل المفتاح للسجل أو للجزء اليساري منه والذي هو موصوف بمستوى أدنى مثل : REC1 بالمثال التالي :

```
ø3  REC - KEY.
    ø5  REC1    PIC xx.
    ø5  REC 2   PIC xxx.
```

ح ـ تعليمة حذف سجل DELETE

الشكل العام للتعليمة :

DELETE file - name RECORD.

تفيد هذه التعليمة في حذف سجل من الملف. و يشترط لتنفيذ هذه التعليمة أن يكون السجل قد قرء بتعليمة القراءة READ كما يجب أن يكون الملف مفتوحاً بشكل INPUT - OUTPUT أي I - O .

كما لاحظنا في تعليمة REWRITE السابقة، فهنا أيضاً نجد أن هذه التعليمة توفر على المبرمج معالجة الملفات وعمل مقارنة بين ملف حركة وملف تعديل من أجل حذف سجل معين وكما مرّ معنا في المثال السابق.

ط ـ تعليمة CLOSE

هذه التعليمة لا تتغير مع هذا النوع من أساليب المعالجة و يبقى الشكل العام :

CLOSE file - name - 1 [,file - name - 2, ...] [WITH LOCK].

وأحيـانـاً يـضاف اليها عبارات خاصة بنوع وسط التخزين المستخدم مثل الاغلاق وعـدم الـسـمـاح لاحد بالتعديل على هذا الملف WITH LOCK وهذه العبارات تختلف باختلاف الحاسب المستخدم.

٨ ـ ٥ ـ ٢ ـ ١ ـ ٣ معالجة الملفات المباشرة بالطريقة المتسلسلة *

يمكـن أن نـقول أن هذا الأسلوب من المعالجة للملفات المباشرة التنظيم DIRECT هـي شـبـيـهـة بـاسـلوب معالجة الملفات المتسلسلة المفهرسة بالطريقة المتسلسلة مع بعض الاختلافات التي سنذكرها :

ـ في قسم ENVIRONMENT DIVISION نجد :

```
** FILE - CONTROL.
        SELECT file - name ASSIGN TO implementor - name
        ORGANIZATION IS DIRECT
        ACCESS MODE IS SEQUENTIAL,
        RECORD KEY IS data - name - 1
        [ALTERNATE RECORD KEY IS data - name - 2]
        [FILE STATUS data - name - 3]
```

نجد أن الاختلاف فقط في عبارة ORGANIZATION DIRECT، أي في اسلوب بناء الملف ـ تنظيم الملف ـ

ـ وفي قسم PROCEDURE DIVISION :

تستخدم التعليمات :

REWRITE, WRITE, READ, OPEN, CLOSE, START, DELETE

(*) قليلة الاستخدام في لغة كوبول على أجهزة غير أجهزة I.B.M

(**) TI990 COBOL MANUALS

كما هي مستخدمة في الملفات متسلسلة التنظيم والواردة في الفقرة ٨ - ٥ - ٢ - ١ - ٢ / أ ، ب ، ج د ، هـ ، ح ، ط ويمكن الرجوع اليها من أجل الشرح.

٨ - ٥ - ٢ - ١ - ٤ معالجة الملفات النسبية RELATIVE FILES بطريقة المعالجة المتسلسلة

يعالج هذا النوع من الملفات النسبية حسب التسلسل المتصاعد ASCENDING لرقم السجل النسبي RELATIVE RECORD NUMBER لكل السجلات الموجودة في الملف.

و يكون التوصيف كالتالي :

ـــ في قسم ENVIRONMENT DIVISION نجد :

```
FILE - CONTROL.
    SELECT file - name, ASSIGN TO device - name
    ORGANIZATION IS RELATIVE
    ACCESS MODE IS SEQUENTIAL,
    RELATIVE RECORD KEY IS data - name - 1
```

حيث أن اسلوب تنظيم الملف هو نسبي ORGANIZATION RELATIVE

وإن اسلوب المعالجة هو متسلسل ACCESS MODE SEQUENTIAL أما RELATIVE RECORD KEY فهو رقم السجل النسبي وهو بمثابة مفتاح، فيجب أن يكون معرّفاً في DATA DIVISION لكن ليس من ضمن تعريف السجل في الملف أي ليس في FD التابع لنفس الملف. يمكن أن يكون في فصل WORKING - STORAGE ولا يجب أن يكون موصوفاً بشكل INDEX أو COMP - 1 أو COMP - 3 ولكن يجب أن يعرف من النوع الرقمي (9).

ـــ في قسم PROCEDURE DIVISION :

نجد أن

أ ـ تعليمة فتح الملف OPEN :

الشكل العام لها

```
        { INPUT  }
OPEN    { OUTPUT }   file - name
        { I - O  }
```

وهو نفس الشكل العام المستخدم في معالجة الملفات المتسلسلة.

ب ـ تعليمة READ :

كذلك أن الشكل العام للتعليمة هو نفس الاشكال السابقة والواردة في اسلوب المعالجة التسلسلية

```
READ  file - name    NEXT    RECORD [INTO identifier]
            [ AT END imperative statement ]
```

جـ ـ تعليمة WRITE :

نفس الشرح الوارد في الفقرة ٨ ـ ٥ ـ ٢ ـ ١ ـ ٢ ـ جـ

والشكل العام :

```
WRITE record name [FROM identifier] INVALID KEY
        imperative statement.
```

د ـ تعليمة REWRITE :

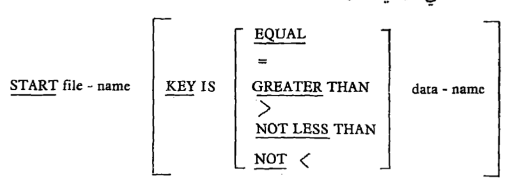
أيضاً كما هو وارد في ٨ ـ ٥ ـ ٢ ـ ١ ـ ٢ د

والشكل العام :

```
REWRITE recordname [FROM identifier].
```

هـ ـ تعليمة START :

كما هي عليه في الفقرة ٨ ـ ٥ ـ ٢ ـ ١ ـ ٢ هـ



ح ـ تعليمة حذف سجل DELETE :

راجع الفقرة ٨ ـ ٥ ـ ٢ ـ ١ ـ ٢ ح

والشكل العام لها :

DELETE file - name RECORD.

ط ـ تعليمة غلق الملفات CLOSE :

والشكل العام لها :

CLOSE file - name - 1 [, file - name - 2 ...].

كما نلاحظ أنه يمكن معالجة الملفات النسبية بصورة بسيطة عند حذف سجل أو عند إضافة سجل جديد على الملف، أو حتى تعديل محتوى السجل. لكن يتم ذلك بطريقة تسلسلية والمبرمج لا يحدد مباشرة السجل الذي يريد العمل عليه إنما عليه القراءة تسلسلياً حتى يصل للرقم المطلوب ثم يجري عليه العمليات اللازمة.

كما ونلاحظ أيضاً أن الملف يعرّف بمفتاح واحد فقط ولا توجد مفاتيح ثانوية ALTERNATE KEYS.

٨ ـ ٥ ـ ٢ ـ ٢ المعالجة العشوائية RANDOM ACCESS (DIRECT ACCESS)

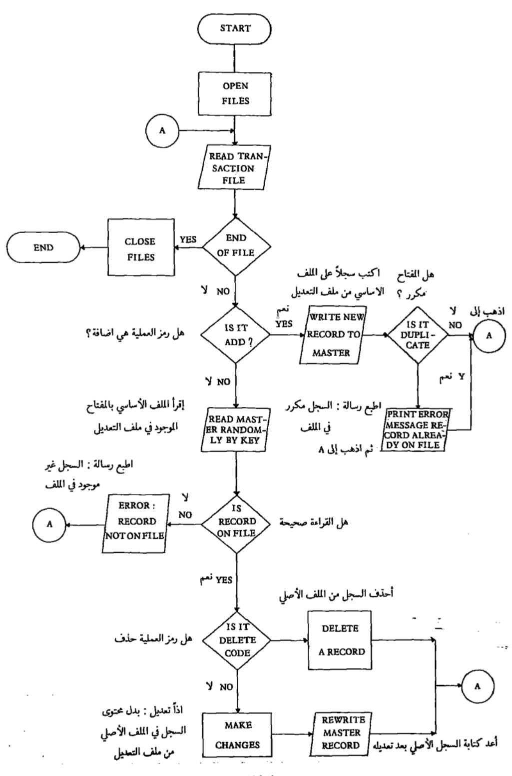
كما علمنا من الفقرات السابقة أن الملف الأساسي MASTER FILE بحاجة دوماً لتحديث UP - TO - DATE حتى تكون المعلومات فيه صحيحة مئة بالمئة.

إن هذه الطريقة من المعالجة والتي تعالج الملفات ذات التنظيم المفهرس INDEXED والمباشر DIRECT والنسبي RELATIVE تسمح بأن تعدل محتوى أي سجل بصورة بسيطة.

وتتم المعالجة غالباً باستخدام ملف تعديلات TRANSACTION FILE على الملف الأساسي MASTER FILE دون الحاجة لملف إخراج جديد للملف الأساسي وكما مرّ معنا في الملفات المتسلسلة التنظيم.

إن ملف التعديل هذا يقرأ من الملف الأساسي بصورة عشوائية دون ترتيب معين و يعالج هذا السجل حسب رمز معين، هذه المعالجة تتضمن تعديل محتوى سجل موجود في الملف الأساسي أو حذف سجل موجود من الملف الأساسي (أحيانا الحذف يكون منطقياً LOGICAL بحيث يوضع بمكان معين في الملف الاساسي رمز يدل على أن هذا السجل محذوف) أو يمكن إضافة سجل جديد على الملف الأساسي بشرط أن يكون هذا السجل غير موجود فيه وإلا تكون عندنا حالة تكرار DUPLICATE وتعالج بعبارة ... INVALID KEY، والشكل التالي يرينا المخطط الانسيابي للمعالجة.

أنظر الصفحة رقم ٥٩٦ لتوصيف شكل المخرجات.


START
OPEN FILES
A
READ TRANSACTION FILE
END OF FILE
YES
CLOSE FILES
END
NO لا
IS IT ADD ?
هل رمز العملية هي اضافة ؟
نعم YES
WRITE NEW RECORD TO MASTER
اكتب سجلاً على الملف الاساسي من ملف التعديل
IS IT DUPLICATE
هل المفتاح مكرر ؟
لا NO
اذهب إلى
A
Y نعم
PRINT ERROR MESSAGE RECORD ALREADY ON FILE
اطبع رسالة : السجل مكرر في الملف
ثم اذهب إلى A
لا NO
READ MASTER RANDOMLY BY KEY
إقرأ الملف الأساسي بالمفتاح الموجود في ملف التعديل
IS RECORD ON FILE
هل القراءة صحيحة
لا NO
ERROR : RECORD NOT ON FILE
اطبع رسالة : السجل غير موجود في الملف
A
نعم YES
IS IT DELETE CODE
هل رمز العملية حذف
DELETE A RECORD
أحذف السجل من الملف الأصلي
A
لا NO
MAKE CHANGES
اذاً تعديل : بدل محتوى السجل في الملف الأصلي من ملف التعديل
REWRITE MASTER RECORD
أعد كتابة السجل الأصلي بعد تعديله

```
*
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT20.
 AUTHOR.      KHAYAT-JNAID.
*
*   يقوم هذا البرنامج بتعديل الملف الاساسي
*              المفهرس من ملف تعديل متسلسل
*
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.    IBM-370.
   OBJECT-COMPUTER.    IBM-370.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
        SELECT INDXMST ASSIGN  TO INDX
        ORGANIZATION   INDEXED;
        ACCESS MODE    RANDOM;
        RECORD KEY     SUB-NO.
        SELECT TRANSF  ASSIGN  TO TRANS
        ORGANIZATION   SEQUENTIAL
        ACCESS MODE    SEQUENTIAL.
        SELECT PRINTF  ASSIGN  TO UT-S-PRFL.
 DATA DIVISION.
 FILE  SECTION.
 FD     INDXMST   LABEL RECORD  IS STANDARD
                        DATA RECORD IS SUB-REC.
 01     SUB-REC.
        03     SUB-NO     PIC  X(5).
        03     SUB-NAME   PIC  X(20).
        03     SUB-ADRES  PIC  X(24).
        03     SUB-DATE.
           05  SUB-YEAR PIC  9(4).
           05  SUB-MNTH PIC  99.
           05  SUB-DAY  PIC  99.
        03     SUB-CODE PIC  99.
        03     SUB-AREA PIC  99.
        03     FILLER   PIC  X(19).
*
 FD     TRANSF     LABEL RECORD  IS STANDARD
             BLOCK CONTAINS 0 RECORDS
                        DATA RECORD IS T-REC.
```

```
 01     T-REC.
        03     T-NO     PIC  X(5).
        03     T-NAME   PIC  X(20).
        03     T-ADRES  PIC  X(24).
        03     T-DATE.
           05  T-YEAR   PIC  9(4).
           05  T-MNTH   PIC  99.
           05  T-DAY    PIC  99.
        03     T-CODEX  PIC  99.
        03     T-AREA   PIC  99.
        03     T-CODE   PIC  99.
        03     FILLER   PIC  X(17).
*
 FD     PRINTF      LABEL RECORD  IS OMITTED
                          DATA RECORD IS  P-REC.
 01     P-REC          PIC      X(132).
*
 WORKING-STORAGE SECTION.
 01     LN1.
  03    FILLER  PIC     X(47)    VALUE SPACES.
  03    FILLER  PIC     X(32)    VALUE
        'خلاصة اخطاء تعديل الملف الاساسي'.
  03    FILLER  PIC     X(47)    VALUE SPACES.
*
 01.    LN2.
  03    FILLER  PIC     X(10)    VALUE SPACES.
  03    FILLER  PIC     X(20)    VALUE
        'رقم المشترك'.
  03    FILLER  PIC     X(20)    VALUE SPACES.
  03    FILLER  PIC     X(20)    VALUE
        'اسم المشترك'.
  03    FILLER  PIC     X(30)    VALUE SPACES.
  03    FILLER  PIC     X(10)    VALUE
        'الملاحظات'.
  03    FILLER  PIC     X(16)    VALUE SPACES.
*
 01     DETAILS.
  03    FILLER  PIC     X(20)    VALUE SPACES.
  03    P-NO    PIC     ZZZ99.
  03    FILLER  PIC     X(25)    VALUE SPACES.
  03    P-NAME  PIC     X(30).
```

```
    03     FILLER   PIC     X(20)     VALUE  SPACES.
    03     P-REMK   PIC     X(16).
    03     FILLER   PIC     X(20)     VALUE  SPACES.
*
  01       FINLIN.
    03     FILLER   PIC     X(51)     VALUE  SPACES.
    03     P-CNTR   PIC     ZZZ9.
    03     FILLER   PIC     X         VALUE  SPACES.
    03     P-RST    PIC     X(8).
    03     FILLER   PIC     X(10)     VALUE
           'عدد السجلات'.
    03     FILLER   PIC     X(51)     VALUE  SPACES.
*
  01       AST      PIC     X(132)    VALUE  ALL  '#'.
*
  01       DASH     PIC     X(132)    VALUE  ALL  '-'.
*
*
*        عدادات تستخدم للتجميع
  01       COUNTERS.
           03 REF-CNTR   PIC     9(4).
           03 ADD-CNTR   PIC     9(4).
           03 UPD-CNTR   PIC     9999.
           03 DEL-CNTR   PIC     9(4).
           03 LN-CNTR    PIC     9(2).
*
 PROCEDURE DIVISION.
         OPEN       INPUT    TRANSF
          I-O       INDXMST
                    OUTPUT   PRINTF.
*        اعطاء القيمة صفر للعدادات
         MOVE       ZEROS          TO    COUNTERS.
         PERFORM N-PAGE THRU  E-PAGE.

*
*        القراءة من ملف التعديل
* T-CODE واختيار رمز العملية
*
 READ-TRANS.
         READ TRANSF AT END
                 PERFORM END-OF-JOB STOP RUN.
```

```
*
       IF T-CODE = '01'
                PERFORM INSRT-RTN THRU EXT-I
       ELSE
       IF T-CODE = '02'
                PERFORM UPDAT-RTN THRU EXT-U
       ELSE
       IF T-CODE = '03'
                PERFORM DELET-RTN THRU EXT-D.

*
                GO TO READ-TRANS.
*
*   الفقرة التالية هي لإضافة سجل جديد
*
 INSRT-RTN.
       PERFORM MOVE-TRNS-MST THRU EXT-MOVE.
       WRITE SUB-REC
             INVALID KEY MOVE
             'السجل مكرر' TO P-REMK
              PERFORM PRINT-LN GO TO EXT-I.
       ADD   1  TO ADD-CNTR.
 EXT-I.
       EXIT.

*
* الفقرة التالية هي لتعديل سجل من الملف الأساسي *
*
 UPDAT-RTN.
       MOVE T-NO TO SUB-NO.
       READ INDXMST INVALID KEY
             MOVE 'السجل غير موجود' TO P-REMK
               PERFORM PRINT-LN GO TO EXT-U.
       PERFORM MOVE-TRNS-MST THRU EXT-MOVE.
       REWRITE SUB-REC INVALID KEY
             MOVE 'السجل غير موجود' TO P-REMK
               PERFORM PRINT-LN GO TO EXT-U.
       ADD   1 TO UPD-CNTR.
 EXT-U.
       EXIT.
```

```
*
* الفقره التاليه هي لحذف سجل من الملف الاساسي *
*
 DELET-RTN.
       MOVE T-NO TO SUB-NO.
       DELETE INDXMST RECORD INVALID KEY
             MOVE 'السجل غير موجود' TO P-REMK
               PERFORM PRINT-LN GO TO EXT-D.
       ADD   1 TO DEL-CNTR.
 EXT-D.
       EXIT.
*
* الفقره التاليه هي لطباعه السجلات التي ليس لها *
* اصل في الملف الاساسي
*
 PRINT-LN.
        ADD     1        TO    REF-CNTR.
        MOVE    T-NO     TO    P-NO.
        MOVE    T-NAME   TO    P-NAME.
        IF      LN-CNTR >     24
                PERFORM N-PAGE THRU E-PAGE.
        MOVE    SPACES   TO    P-REC.
        WRITE   P-REC    FROM  DETAILS.
        WRITE   P-REC    FROM  DASH.
        ADD     1        TO    LN-CNTR.
 EXT-LN.
        EXIT.
*
*    الفقره التاليه هي لنقل الحقول من ملف الحركه
*                                 الى الملف الاساسي
*
 MOVE-TRNS-MST.
       MOVE  T-NO     TO SUB-NO.
       MOVE  T-NAME   TO SUB-NAME.
       MOVE  T-ADRES  TO SUB-ADRES.
       MOVE  T-DATE   TO SUB-DATE.
       MOVE  T-CODEX  TO SUB-CODE.
       MOVE  T-AREA   TO SUB-AREA.
 EXT-MOVE.
       EXIT.
```

```
*
الفقره التاليه هى لطلب الصفحه وكتابه العناوين *
*
 N-PAGE.
      MOVE   SPACES TO     P-REC.
      WRITE P-REC   AFTER PAGE.
      WRITE P-REC   FROM   LN1 AFTER 2 LINES.
      WRITE P-REC   FROM   LN2 AFTER 3 LINES.
      WRITE P-REC   FROM   AST AFTER 1 LINE.
      MOVE   SPACES TO     P-REC.
      MOVE   ZERO   TO     LN-CNTR.
 E-PAGE.
      EXIT.
*
*      الفقره التاليه هى لطباعه المجاميع النهائيه
*
 END-OF-JOB.
        PERFORM  N-PAGE      THRU  E-PAGE.
        MOVE    'المضافه'  TO      P-RST.
        MOVE    ADD-CNTR    TO      P-CNTR.
        WRITE   P-REC       FROM    FINLIN AFTER 20.
        MOVE    'المعدله'  TO      P-RST.
        MOVE    UPD-CNTR    TO      P-CNTR.
        WRITE   P-REC       FROM    FINLIN AFTER 2.
        MOVE    'المحذوفه' TO      P-RST.
        MOVE    DEL-CNTR    TO      P-CNTR.
        WRITE   P-REC       FROM    FINLIN AFTER 2.
        MOVE    'المرفوضه' TO      P-RST.
        MOVE    REF-CNTR    TO      P-CNTR.
        WRITE   P-REC       FROM    FINLIN AFTER 2.
        WRITE   P-REC       FROM    AST.
*
        CLOSE   INDXMST, TRANSF, PRINTF.
 EXT-END.
        EXIT.
*
```

ملاحظة : إن حاصل التعديل بهذه الطريقة هو مساوي ومطابق تماماً لحاصل التعديل في المعالجة المتسلسلة إنما طريقة العمل مختلفة .

على أن هذه ليست الطريقة الوحيدة في المعالجة ؛ انما يمكن التعديل أن يكون مباشرة من الشاشة الى الملف دون الحاجة الى ملف تعديل ؛ وفيه تظهر شاشة أولية يعطى فيها خيار إما لحذف أو لاضافة أو لتعديل سجل ، بعدها يطلب رقم السجل المراد العمل عليه و يدخل عن طريق الشاشة و يستدعى الى الشاشة و يظهر محتواه ثم تنفذ العملية المطلوبة ، اما في حالة الاضافة فتدخل البيانات مباشرة من الشاشة الى الملف الموجود على قرص ممغنط .

ويمكن تتبع المخطط المنطقي السابق في المعالجة عن طريق الشاشة إنما عوضاً عن القراءة من ملف تعديل يطلب القراءة من الشاشة فقط .

سنستعرض بالتفصيل المعالجة العشوائية لكل نوع من أنواع تنظيم الملفات :

٨ ـ ٥ ـ ٢ ـ ٢ ـ ١ معالجة الملفات المتسلسلة بالطريقة العشوائية :

لا يمكن معالجة الملفات ذات التنظيم المتسلسل SEQUENTIAL ORGANIZATION بالطريقة العشوائية RANDOM ACCESS وذلك لعدم وجود علاقة بين مفتاح السجل ووسط التخزين .

٨ ـ ٥ ـ ٢ ـ ٢ ـ ٢ معالجة الملفات المفهرسة المتسلسلة بالطريقة العشوائية :

نجد أنه في قسم

```
ENVIRONMENT DIVISION
:
FILE - CONTROL.
      SELECT implementor - name ASSIGN TO
      ORGANIZATION IS INDEXED,
      ACCESS MODE IS RANDOM,
```

RECORD KEY IS data - name - 1,

[ALTERNATE KEY IS data - name - 2, [WITH DUPLICATES]

[FILE STATUS IS data - name - 3].

مفتاح السجل RECORD KEY :

يجب تعيين اسم الحقل الذي يعتبر مفتاحاً. واسم الحقل هذا يجب أن يرد توصيفه في توصيف الملف في DATA DIVISION ، و يعتبر هذا المفتاح هو الذي يميز سجلا عن آخر ولا يجوز أن يكون مكرراً في الملف و يسمى PRIME KEY أي المفتاح الأ ولي. كما لا يجوز تعديل هذا المفتاح بتغيير الرقم، وإن اضطر المبرمج لأن يعدل المفتاح فما عليه إلا أن يحذف السجل ثم يضيفه برقم جديد.

المفتاح الملحق ALTERNATE KEY* :

هذه العبارة تعرف المفتاح الثانوي، الثالثي، الرابعي ... للملف؛ إذ يمكننا أن نتعرف على السجل بأكثر من مفتاح، فمثلاً سجل الموظف يمكن أن نتعرف عليه برقمه كمفتاح اساسي أولي أو باسم الموظف كمفتاح ثانوي، أو بمرتبته أي مفتاح ثالث أو برقم حفيظته وهو مفتاح رابع ....

فاذا كان من الممكن أن يتكرر المفتاح الثانوي أو الثالثي ... في الملف كأن يكون أكثر من موظف له نفس الاسم الثلاثي، أو أن يكون أكثر من موظف بنفس المرتبة فنضيف عبارة ـ WITH DUPLICATES ـ وهي تعرف الحاسب أن هذا المفتاح غير الأ ولي مكرّر في الملف، أما اذا لم نذكر هذه العبارة WITH DUPLICATES أمام توصيف المفتاح الثانوي فهذا يعني أن هذا المفتاح الثانوي وحيد أيضاً.

(*) يمكن أن يصل عدد المفاتيح الثانوية الى ثلاثة عشر مفتاحاً في حاسب TI990 وتختلف باختلاف الحاسب المستخدم.

أما عبارة حالة الملف FILE STATUS فتشير الى حالة الملف ويجب أن نحجز مساحة في WORKING - STORAGE حتى نعرف data - name - 3 ويجب أن تكون رقمية و بطول حرفين (2 NUMERIC BYTES)؛ إن هذا الحقل يتبدل حسب تبدل حالة الملف فيوضع فيه أرقام لها معنى في كل حاسب، لذلك يرجع الى مراجع كل حاسب من أجل معرفة معنى هذه الأ رقام.

فلو كانت القراءة ناجحة مثلاً لكان محتواه صفراً؛ أما في حالة عدم وجود المفتاح في الملف فيتوضع فيه قيمة يدل المرجع على أن السجل غير موجود. وهكذا بالنسبة لكل معالجة تطرأ على الملف.

ـ في قسم الاجراءات PROCEDURE DIVISION

نستعرض فيما يلي التعليمات والشكل العام لكل منها مع شرح لها :

أ ـ تعليمة OPEN :

```
        { INPUT    file - name - 1 ...  }
OPEN    { OUTPUT   file - name - 3 ...  }
        { I - O    file - name - 4 .... }
```

كما مرّ معنا في الشرح السابق الفقرة ٨ ـ ٥ ـ ٢ ـ ١ ـ ٢ فقرة أ.

ب ـ تعليمة القراءة READ :

الشكل العام لها :

```
READ file - name RECORD [INTO identifier] [KEY IS data - name]
        [INVALID KEY imperative - statement].
```

تـستخدم هذه التعليمة من أجل القراءة من ملف ذي تنظيم مفهرس وبمعرفة المفتاح data - name وهـو نـفـسـه المـوصف في FD للملف ، وكذلك في SELECT ومحتوى هذا المـفـتـاح يـأتي إما عن طريق ملف آخر مسلسل sequential هو ملف تعديل مثلاً أو عن طريق الشاشة بتعليمة ACCEPT .

وفي حـالـة نـجـاح القراءة ـ كان السجل موجوداً ـ ينتقل التنفيذ للتعليمة التالية لـ READ وإلا يذهب التنفيذ الى التعليمات التي بعد عبارة INVALID KEY .

مثال :

```
READ TRNSF AT END GO TO E - O - JOB.
MOVE T - NO TO MST - NO.
READ MSTFILE RECORD KEY IS MST - NO
INVALID KEY GO TO ERR - RTN.
MOVE ....
ADD ....
```

نلاحظ في المثال : أنـه في البـداية قرأنا من ملف التعديل بصورة متسلسلة (لأن تـنـظـيـمـه مـتـسـلـسل مثلاً) ثم حركنا محتوى المفتاح T - NO الموجود في ملف التعديل والـذي نـريـد أن نـعـدل نـظـيـره في المـلـف الأسـاسي ؛ نـحرك محتواه الى مفتاح الملف الأسـاسي MST - NO ، ونقرأ الملف الأساسي MSTFILE بصورة عشوائية RANDOM (دون أي تـرتـيـب أو مـقـارنـة ـ انـظر المعالجة المتسلسلة) فاذا تمت القراءة بنجاح انتقل الـتـنـفـيـذ للـتـعليمة التالية لـ READ وهي MOVE ثم ADD ، أما إذا كان هذا الرقم T - NO غير موجود في الملف الأساسي فتتشكل حالة INVALID KEY والتنفيذ يذهب الى التعليمة GO TO ERR - RTN وهناك يظهر أو يطبع رسالة حسب المطلوب .

مثال ٢ :

```
ACPT.
    ACCEPT MST - NO.
    READ MSTFILE RECORD KEY
    IS MST - NO INVALID KEY
        DISPLAY "RECORD IS NOT EXIST"
      GO TO ACPT.
      DISPLAY "RECORD IS EXIST"
          ⋮
```

نلاحظ في المثال الثاني : ان محتويات المفتاح قد أخذناه عن طريق الشاشة بتعليمة ACCEPT ، ونقرأ فيه الملف؛ فاذا كانت القراءة ناجحة أي الملف موجود ظهرت عبارة RECORD IS EXIST ، أما اذا كانت القراءة غير ناجحة فتنشأ حالة INVALID KEY وتظهر رسالة "RECORD IS NOT EXIST" وتذهب المعالجة الى ACPT لقبول سجل جديد، وهكذا ....

جـ- تعليمة الكتابة **WRITE**

الشكل العام :

WRITE record - name [FROM identifier] [INVALID KEY imprative - statement].

تستخدم كما اسلفنا في اساليب المعالجة المتسلسلة والعشوائية للملفات ذات التنظيم المفهرس والعشوائي والنسبي، هذه التعليمة تكتب على ملف المعلومات المنقولة من ملف آخر أو عن طريق الشاشة.

وحالة INVALID KEY تنشأ هنا عندما يكون محتوى حقل المفتاح RECORD KEY له مماثل في الملف الاساسي المراد الكتابة عليه، أي حالة تكرار

المفتاح على نفس الملف. ويجب ان تنفذ هذه التعليمة بعد فتح الملف كملف اخراج OUTPUT أو ملف ادخال ـ اخراج I - O كما يجب أن نذكر أنه في حال وجود مفتاح ثانوي وغير معرّف بعبارة WITH DUPLICATES أي لا يقبل التكرار وأضفنا سجل مفتاحه الاساسي غير مكرر، أما الثانوي فهو مكرر لكن دون ذكر العبارة السابقة WITH DUPLICATES فإن حالة INVALID KEY تتشكل أيضاً، لذلك يجب الانتباه ليس للمفتاح الاساسي فقط بل والثانوي أيضاً.

مثال :

```
WRITE ENDX - REC INVALID KEY GO TO ERR - RTN.
PERFORM TOP - DOWN THRU EXT - TOP.
     :
     :
```

في هذا المثال نجد أنه اذا نفذت تعليمة WRITE بنجاح ينتقل تسلسل التنفيذ الى تعليمة PERFORM وإلا تتشكل حالة INVALID KEY و ينتقل التنفيذ الى تعليمة GO TO ERR - RTN وفيها يعالج السجل حسب المطلوب.

د. تعليمة اعادة الكتابة REWRITE :

الشكل العام لها :

```
REWRITE record - name [FROM identifier]
        [INVALID KEY imperative - statement].
```

تستخدم هذه التعليمة لتثبيت تعديل سجل معين؛ وتتشكل حالة INVALID KEY عندما يكون محتوى المفتاح الأولي PRIME KEY غير مطابقة لأي قيمة في الملف الأساسي ـ السجل غير موجود.

مثال :

```
READ - LOOP.
    READ TFILE AT END GO TO E - O - PROCESS.
    MOVE T - NO TO MST - NO.
    READ MSTFILE INVALID KEY GO TO PRT - ERR.
    MOVE T - NAME TO MST - NAME.
    MOVE T - ADRESS TO MST - ADRESS.
    |
    |
    |
   REWRITE MST - REC INVALID KEY GO TO PRT - ERR.
   GO TO READ - LOOP.
PRT - ERR.
```

نلاحظ من المثال السابق أنه تمت القراءة من ملف التعديل ثم حُرِّكَ المفتاح من ملف التعديل الى مفتاح الملف الأساسي.

بعدها تمت القراءة من الملف الأساسي في حال عدم نجاح القراءة يذهب التنفيذ الى فقرة PRT - ERR وإلا يتابع عمله بتحريك الحقول من ملف التعديل الى حقول الملف الأساسي بعدها يُثبت التعديل بتعليمة اعادة كتابة السجل REWRITE، فاذا كان محتوى المفتاح الاساسي قد غير أثناء المعالج فتتشكل عندنا حالة INVALID KEY وإلاّ يتابع التنفيذ الى تعليمة GO TO READ - LOOP ... وهكذا.

هـ - تعليمة START :

لا تنفذ مع اسلوب المعالجة العشوائية.

ح ـ تعليمة الحذف DELETE :

الشكل العام :

DELETE file - name [RECORD] [INVALID KEY imperative - statement].

تستخدم هذه التعليمة لحذف سجل من الملف ويمكن استخدام مكان هذا السجل لاضافة سجل آخر، ويمكن للسجل أن يحذف منطقياً LOGICALY أي نظرياً ، لكن بالواقع يكون موجود ولكن لا يمكن معالجته ، وفي كلا الحالتين يمكن للحاسب استخدام مكانه لتسجيل سجل آخر.

ويجب أن ننوه أنه من مقتضيات المعالجة أحياناً يلزم أن نحذف سجلاً ـ نوقف معالجته ـ لفترة معينة ثم نعاود إضافته للملف ومعالجته ، فاذا استخدمنا تعليمة DELETE فنجد أن محتوى السجل قد فقد كلياً ؛ لذا نستخدم حقلاً معيناً ضمن السجل ونعطيه رمزاً ونضع في هذا الحقل رمز الحذف وبالتالي نقارن عليه فاذا كان رمز الحذف موجوداً يعني أنه محذوف مؤقتاً .

مثال :

لو أخذنا رمز الحذف المؤقت هوT والحذف الدائم هوD في ملف التعديل وحقل تثبيت الحذف في الملف الأساسي هوMST - STS يكون :

```
            READ TFILE AT END GO TO E - O - JB.
            MOVE T - NO TO MST - NO.
            IF T - STS  = "D"
حذف حقيقي    DELETE MST - FILE RECORD INVALID KEY
PHISICAL DELETION           GO TO ERR - RTN.
            MOVE T - NO TO MST - NO.
            READ MSTFILE INVALID KEY GO TO ERR - RTN.
```

```
MOVE "T" TO MST - STS.
REWRITE MST - REC INVALID KEY GO TO ERR - RTN.
```
حذف نظري

LOGICAL DELETION

تتم قراءة ملف التعديل، فاذا كان الرمز D يعني أنه مطلوب حذف السجل نهائياً من الملف الأساسي لذلك تحرك قيمة المفتاح ويحذف بموجبه السجل نهائياً من الملف بتعليمة DELETE أما إذا لم يكن الرمز D وكان T فهذا يعني أنه حذف مؤقت فيحرك المفتاح ثم يقرأ السجل المطلوب حذفه حذف مؤقت تحرك قيمة T (دالة على الحذف المؤقت) الى الملف الأساسي و يعاد كتابة السجل من جديد بتعليمة REWRITE، وفي أثناء المعالجة يعمل تحقق على الحقل MST - STS فاذا كانت التعليمة T موجودة فهذا يعني أنه محذوف مؤقتاً فلا يعالج، أما اذا كان غير ذلك فيعالج.

وعند الانتهاء من توقيف هذا السجل عن المعالجة يحرك للحقل MST - STS القيمة SPACE و يعاد كتابة السجل من جديد بتعليمة REWRITE فيرجع السجل كما كان في السابق و يكون تحت المعالجة.

ط - تعليمة CLOSE :

لها الشكل العام ... ,CLOSE file - name - 1

٨ - ٥ - ٢ - ٢ - ٣ معالجة الملفات المباشرة بالطريقة العشوائية :

نجد أنه يمكن معالجة الملفات ذات التنظيم المباشر DIRECT بالطريقة العشوائية RANDOM وذلك حسب الوصف التالي :

— في قسم ENVIRONMENT DIVISION :

```
FILE - CONTROL.
    SELECT file name ASSIGN TO ...
```

```
ORGANIZATION IS DIRECT,
ACCESS MODE IS RANOM
RECORD KEY IS data - name - 1,
[ALTERNATE RECORD KEY IS data - name - 2]
[WITH DUPLICATES]
[FILE STATUS IS data - name - 4].
```

يرجع للفقرة ٨ ـ ٥ ـ ٢ ـ ٢ ـ ٢ من أجل شرح عبارة RECORD KEY ، ALTERNATE RECORD KEY ، WITH DUPLICATES , FILE STATUS .

والاختلاف الوحيد هو في عبارة ORGANIZATION IS DIRECT أما بالنسبة للتعليمات فهي كما وردت في الفقرة ٨ ـ ٥ ـ ٢ ـ ٢ ـ ٢ أ ، ب ، جـ ، د ، هـ ، ح ، ط .

والاختلاف في المعالجة هو ضمني ويخص الحاسب إذ يتم البحث عن السجل في هذه الطريقة من المعالجة بوساطة عملية حسابية معقدة RANDOMIZATION و يصل للعنوان المطلوب و يقرأ السجل فاذا لم يجده تتشكل حالة INVALID KEY بينما في المعالجة المفهرسة يتم البحث عن طريق ملف ملحق يدعى الفهرس INDEX و يأخذ العنوان منه ليقرأ ملف البيانات .

كما قلنا ان طريقة المعالجة ضمنية ولا تهم المبرمج في لغة كوبول في شيء .

٨ ـ ٥ ـ ٢ ـ ٢ ـ ٤ معالجة الملفات النسبية بالطريقة العشوائية

كما مرّ معنا سابقاً إن الملفات النسبية ذات ترتيب متسلسل بالنسبة لبداية الملف ؛ فان التوصيف يكون على الشكل

ـ في قسم ENVIRONMENT DIVISION :

```
*   FILE - CONTROL.
        SELECT file ASSIGN TO ...
        ORGANIZATION IS RELATIVE
        ACCESS MODE IS RANDOM,
        RELATIVE KEY IS data - name - 1
        [FILE STATUS IS data - name - 2.]
```

كما نلاحظ هنا فانه لا يوجد لدينا RECORD KEY إنما يستعاض عنها بـ RELATIVE KEY بالمفتاح النسبي وهو رقم السجل ضمن الملف ولا يكون موصفاً في FD التابع للملف file name نفسه.

كما أنه لا يوجد هنا ALTERNATE KEY مفتاح ثانوي، لأن اسلوب التنظيم هنا بأن يعرّف كل سجل حسب تسلسل موصفه بالنسبة لبداية الملف.

إن تسلسل المعالجة هنا يحدده المبرمج فهو الذي يطلب سجلا معينا للتعديل أو الحذف وذلك بوضع قيمة المفتاح في الحقل المسمى RELATIVE KEY بعد ذلك تتم المعالجة وفق هذه القيمة.

ـ في قسم الاجراءات PROCEDURE DIVISION

نجد أن الكلام الذي ينطبق على معالجة الملفات ذات التنظيم المفهرس بالطريقة العشوائية ينطبق على أسلوب المعالجة للملفات ذات التنظيم النسبي وذلك للتعليمات الواردة في الفقرة ٨ - ٥ - ٢ - ٢ - ٢ ضمن أ، ب، جـ، د، حـ، ط لذلك يرجع لتلك التعليمات من أجل معرفة استخدامها.

(*) TI COBOL MANUALS

عدا تعليمة START التي لا تستخدم مع اسلوب المعالجة العشوائية RANDOM ACCESS.

انظر التمارين الملحقة في هذا الفصل من أجل استخدام أنواع اساليب المعالجة المختلفة.

٨ - ٥ - ٢ - ٣ المعالجة الحركية DYNAMIC ACCESS

في كثير من الأحيان يضطر المبرمج لأن يستخدم كلاً من اسلوبي المعالجة المتسلسلة والعشوائية في برنامج واحد لمعالجة ملف واحد لذلك أوجدت طريقة المعالجة الحركية والتي فيها يستطيع المبرمج أن يعالج ملفاً معيناً باسلوبي معالجة متسلسلة وعشوائية. والفكرة تتضح أكثر بالمثال التالي :

لو فرضنا أنه لدينا ملف يحتوي على اسماء طلاب الجامعات في المملكة ولو فرضنا أنه لدينا ١٠ جامعات وكل جامعة فيها عدد معين من الكليات وفي كل كلية هناك نظام السنوات وكل سنة عدد من الطلاب. فلو طلب منا معلومات معينة عن الجامعة رقم ٨ مثلاً ؛ فلو اتبعت الطريقة المتسلسلة في المعالجة SEQUENTIAL ACCESS للزم الأمر أن نقرأ جميع طلاب الجامعات السبع حتى نصل للجامعة رقم ٨ لبدء المعالجة وهذا يستغرق وقتاً طويلاً.

ولو اتبعنا طريقة المعالجة العشوائية بالوصول للجامعة رقم ٨ لما أمكننا متابعة العمل على هذه الجامعة لأن هذا الأسلوب يوصلنا لأول سجل في الجامعة رقم ٨ ولا نستطيع أن نتابع المعالجة من مكاننا هذا.

هنا نجد أن استخدام اسلوب المعالجة الحركي يسمح للمبرمج أن يصل لسجل معين وأن يبدأ منه المعالجة المتسلسلة ففي مثالنا السابق يمكن أن نصل الى الجامعة رقم ٨ مباشرة بتعليمة START وأن نبدأ المعالجة منها بتعليمة ....READ NEXT حتى ننتهي من جميع طلاب الجامعة المطلوبة ثم ننتقل لغيرها وهكذا ....

واسلوب المعالجة الحركي يمكن أن يأتي مع كل أنواع تنظيم الملفات، ويمكن لجميع التعليمات السابقة أن تستخدم فيه، لذلك يرجع للفقرة ٨ - ٥ - ٢ - ٢ من أجل استخدام التعليمات المختلفة وهي :

```
أ   OPEN  { INPUT  }  file - name - 1,
          { OUTPUT }  file - name - 2,
          { I - O  }  file - name - 3, ...

ب  { READ file - name } NEXT RECORD AT END imperative - statement
   { READ file - name } INVALID KEY imperative - statement

ج  { WRITE Record - name.
   { WRITE Record - name    } INVLAID KEY imperative - statement

د   START file - name KEY IS { EQUAL TO                        }
                             { GREATER THAN       data - name  }
                             { NOT LESS THAN                   }
         [ INVALID KEY imperative - statement ]

هـ  REWRITE record - name [INVALID KEY imperative - statement ]

و   DELETE file - name RECORD [INVALID KEY imperative - statement]

ط   CLOSE file - name, ... [WITH LOCK].
```

و يكون شكل SELECT هو التالي :

```
SELECT file - name ASSIGN TO implementor - name,
              ORGANIZATION IS   { SEQUENTIAL }
                                { INDEXED    }
                                { DIRECT     }
                                { RELATIVE   }
```

ACCESS MODE IS DYNAMIC,

الجدول التالي يبين لنا التعليمات المسموح استخدامها حسب نوع المعالجة، ونوع الملف المستخدم.

نوع المعالجة File Access Mode (*)	التعليمات Statement	نوع تعليمة فتح الملف Open Mode — ادخال INPUT	اخراج OUTPUT	ادخال اخراج I - O
Sequential	READ	×		×
متسلسلة	WRITE		×	
	REWRITE			×
	START	×		×
	DELETE			×
Random	READ	×		×
عشوائية	WRITE		×	×
	REWRITE			×
	START			
	DELETE			×
Dynamic	READ	×		×
حركية	WRITE		×	×
	REWRITE			×
	START	×		×
	DELETE			×

(*) TI990 COBOL MANUALS

تظهر علامة × التعليمة المسموح استخدامها مع نوع فتح الملف مثال :

في نوع المعالجة العشوائية RANDOM تعليمة READ مثلاً تستخدم عند فتح الملف أما كـ INPUT أو I - O أو ادخال ـ اخراج .

نلاحظ أن في نوع المعالجة الحركية DYNAMIC أن كل التعليمات يمكن أن تستخدم عند فتح الملف كـ ادخال اخراج I - O .

٨ ـ ٦ اسئلة وتمارين

سؤال ١ ـ لدينا ملف للزبائن اسمه CUSTFILE يحتوي على سجلات CUSTREC والسجل يشمل رقم الزبون CUSTNO ويمثل مفتاح السجل KEY وبيانات أخرى عن الزبون مثل اسمه وعنوانه وحالته المالية . فاذا أردنا كتابة برنامج يساعد في معرفة الزبون الجديد إذا كان موجود سابقاً أو لا . وإذا لم يكن موجوداً إضافة سجل جديد على الملف ، فما هي العبارات والتعليمات التي يجب ذكرها في البرنامج والتي تشمل النواحي التالية علماً بأن الملف منظم كملف مفهرس ، وفق رقم الزبون .

أ ـ عبارة SELECT لتداول الملف CUSTFILE

ب ـ العبارات أو التعليمات اللازمة لاضافة زبون جديد على الملف مع الإشارة الى هذه الإضافة على وحدات الاخراج (شاشة أو طابعة) .

سؤال ٢ ـ لدينا الملف DATEFILE الموصوف في قسم البيانات كما يلي :

```
FD   DATEFILE ...
φ1   DATREC
       φ3   YEAR - MONTH.
         φ5   YY     PIC XX.
```

```
        ϕ5    MM      PIC 99.
    ϕ3    ETC - FLD     PIC X (76).
```

وهذا الملف مفهرس حسب الحقل YEAR - MONTH ، والمطلوب :

أ - كتابة التعليمات اللازمة لقراءة هذا الملف اعتباراً من السنة رقم 79 ثم نقل التسلسل الى الفقرة END - PROCESS عندما تتبدل السنة 79 .

ب - ما هي صيغة التداول ACCESS التي سوف توصف في عبارة SELECT .

سؤال ٣ - أشر الى العبارة الصحيحة بـ (✓) وإلى الخطأ بـ (×) لكل من العبارات التالية :

١ - يُنشأ عادة الملف المفهرس Indexed file مع طريقة تداول عشوائية ACCESS IS RANDOM ولكنه يُقرأ مع طريقة تداول متسلسلة ACCESS IS SEQUENTIAL .

٢ - إذا وردت العبارة ACCESS IS SEQUENTIAL مع عبارة SELECT عندئذ يمكن معالجة الملف الموصوف بطريقة عشوائية Randomly اذا كان سجله يحتوي على مفتاح KEY .

٣ - تعليمة REWRITE يمكن أن تأتي فقط مع الملف الذي تم فتحه على أساس I - O .

٤ - يمكن استخدام جملة INVALID KEY مع تعليمة READ أو تعليمة WRITE .

٥ - إن الملفات المخزنة على الأقراص الممغنطة لا يمكن معالجتها تسلسلياً Sequentially أبداً .

٦ - يرد مع تعليمة DELETE اسم السجل المراد حذفه من الملف .

٧ - تستخدم تعليمة SET لأجل تخزين قيمة الدليل Index وتحديد قيمته .

٨ - يرد تعريف ملف الفرز في قسم البيانات DATA DIVISION وفي الفصل الخاص بالملفات FILE SECTION تحت اسم SD .

٩ - إن الملف الموصوف كملف مفهرس Indexed لا يحتاج الى عملية فرز SORT على المفاتيح (المفتاح) المعرّفة في عبارة SELECT لأنه يُرتب آلياً .

١٠ - حذف سجل من الملف النسبي RELATIVE بتعليمة DELETE يؤدي الى وجود فراغ مكان السجل المحذوف و بالتالي يمكن كتابة سجل آخر جديد يحمل نفس رقم المفتاح مكانه .

تمرين ١ :

لدينا ملفان موجودان على شريط ممغنط مفروزان تصاعدياً على رقم السلعة ؛ الملف الأول يحتوي على سجلات ذات أرقام السلع المفردة . والملف الثاني يحتوي على سجلات ذات أرقام السلع المزدوجة ؛ المطلوب منك كتابة برنامج يدمج ملفي الادخال في ملف اخراج واحد اسمه OUTMRG علماً بأن شكل السجلات كالتالي :

الملف الأول (الأرقام المفردة) : INFILE 1

```
IN1 - NO     PIC     9 (5)
IN1 - DATA     PIC     X (35)
```

الملف الثاني (الأرقام المزدوجة) : INFILE 2

```
IN2 - NO     PIC     9 (5)
IN2 - DATA     PIC     X (35)
```

ملف الإخراج : OUTMRG

```
OUT - NO     PIC     9 (5)
OUT - DATA     PIC     X (35)
```

مفتاح الحل :

١ ـ أقرأ السجلين الأول والثاني.

٢ ـ قارن مفتاحي السجلين IN1 - NO و IN2 - NO وانقل سجل الرقم الأصغر الى ملف الاخراج واكتبه.

٣ ـ اعد القراءة من الملف الذي يحتوي على الرقم الأصغر ـ المنقول سابقاً ـ.

٤ ـ استخدم HIGH - VALUES من أجل معالجة نهايات الملفات.

الفصل التاسع

البرمجة الهيكلية بلغة كوبول

٩ ـ ١ لمحة عامة

يكون هدف المبرمج المبتدىء بلغة كوبول كتابة برنامج بسيط يعمل بدون خطأ ويمكن أن يسمى Working Program ولكن بعد فترة من الممارسة يصبح الغرض الأول للمبرمج كتابة برنامج فعال وكفؤ يمكن تسميته Efficient Program ويمتاز هذا النوع الآخر بسهولة قراءته ومراجعته وامكانية تطويره وتعديله من قبل مبرمج آخر غير المبرمج الذي قام بكتابته. وبذلك يكون الهدف من البرمجة بلغة كوبول باختصار: برنامجاً سهل القراءة والمراجعة وسليماً ومختبراً بشكل جيد من الناحية المنطقية ليؤدي الغرض الذي كتب من أجله.

وعلى مر الزمن بُدىء في تجميع أساليب تقنية وطرق من أجل التوصل الى طريقة لكتابة برنامج يحقق الأهداف المذكورة آنفاً. وسوف نقدم فقط ضمن هذا المؤلف فكرة عن البرمجة الهيكلية Structured Programming كوسيلة مساعدة في لغة كوبول للوصول الى برنامج سهل القراءة وذي منطق سليم. وكذلك سنتعرض الى نقاط قياسية* أثبتت الخبرة الطويلة في برمجة كوبول وجوب وجودها في البرنامج السليم المنشود. وهذه النقاط تساعد على الوصول الى برنامج سهل القراءة والمراجعة والتطوير في المستقبل دون تعقيد، وليست ملزمة للمبرمج المبتدىء.

(*) يقصد بها Coding Standard in COBOL

٩ ـ ٢ نظرية البرمجة الهيكلية STRUCTURED PROGRAMMING

قد يكون المبرمج يتبع في برنامجه عناصر البرمجة الهيكلية المنطقية منذ مدة ولكنه لم يلتفت الى تصنيفها على النحو الذي وضعت به هذه الهياكل المنطقية في نطاق نظرية تدعى نظرية البرمجة الهيكلية والتي تهدف الى وضع نظام من أجل كتابة البرنامج السهل والمنطقي. وقد احتوت هذه النظرية* على ثلاثة هياكل منطقية أساسية وهي: التسلسل Sequence والاختيار Selection والدوارة Iteration كما هو مبين في الأشكال الموضحة برسم المخطط الانسيابي Flowchart **.

وهيكل التسلسل يؤكد على تنفيذ التعليمات ضمن البرنامج بشكل متتالي، وهذا هو الشكل الطبيعي لأي برنامج ما لم يتم نقل التسلسل من قبل المبرمج. فمثلا العملية A يجب أن تنفذ قبل B كما هو في المخطط الانسيابي.

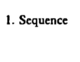
أما هيكل الاختيار فهو عبارة عن مقارنة بين فعلين أو عملين واختبار شرط تحقق أحدهما يسمى Predicate أو Test. فاذا كانت نتيجة الشرط صحيحة True يتم تنفيذ العملية A واذا كانت النتيجة غير صحيحة False يتم تنفيذ العملية B.

أما الدوارة فهي عبارة عن تكرار تنفيذ عملية معينة عدد من المرات طالما أن شرط نهايتها صحيح True واذا أصبح هذا الشرط غير صحيح فتكون العملية قد توقفت عن التكرار. فالعملية A سوف تتكرر عدداً من المرات طالما أن الشرط P صحيح. وعندما يصبح هذا الشرط غير صحيح False تتوقف تنفيذ هذه العملية. و ينتقل البرنامج الى العملية أو التعليمة التالية.

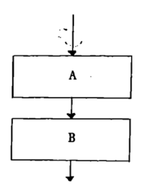
و يلاحظ من أشكال البرمجة الهيكلية أنها تأخذ مدخلا واحداً One Entry ومخرجاً واحداً One Exit لكل شكل من عناصرها، وهذا أمر ضروري لسهولة تتبع خطوات البرنامج.

(*) Techniqunes of Program Structure and Design, Edward Yourdon, Prentice - Hall, 1975, PP 145 - 147.

(**) هناك عناصر ثانوية أخرى ورموز خاصة بهذه الطريقة لا سبيل لذكرها في هذا المؤلف.

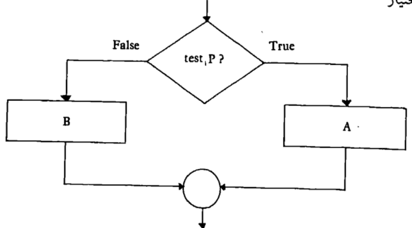


1. Sequence — ١ ـ التسلسل

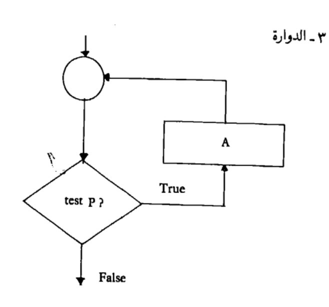




2. Selection — ٢ ـ الاختيار

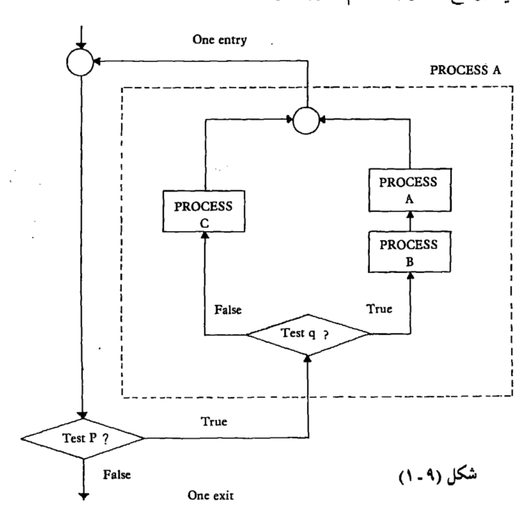


3. Iteration — ٣ ـ الدوارة



كـمـا أن هـذه الأشكـال الأسـاسية يمكن أن تجتمع في برنامج واحد بحيث تكون العـملية A تحتوي على تسلسل واختيار كما هوموضح في الرسم التالي لمخطط انسيابي. (الشكل رقم ٩-١).

و يـلاحـظ أن هـذه الأشكال لا تحتوي على ما يسمى بلغة كوبول بتعليمة GO TO ليـس لأن هـدف البرمجة الهيكلية كتابة برامج بدون GO TO بل لأنها تزيد في صعوبة تـتبع البرنامج ومعرفة خطواته المنطقية وهذا يتعارض مع سهولة القراءة والفهم المطلوبين في البرنامج المقصود باستخدام اسلوب البرمجة الهيكلية.



شكل (٩-١)

٩ ـ ٣ أشكال البرمجة الهيكلية بلغة كوبول :

إن هيكل التسلسل مطبق أصلا في لغة كوبول حيث أن الوضع الطبيعي لتنفيذ تعليمات البرنامج على التتالي. أما هيكل الاختيار والمسمى IFTHENELSE ينطبق بوساطة تعليمة IF ... ELSE وكذلك هيكل الدوارة والمسمى DOWHILE يطبق بتعليمة PERFORM ... UNTIL.

الاختيار : و يكتب في شكله العام :

```
IF Condition
        Statement - 1
ELSE
        Statement - 2.
```

حيث أن Statement - 1 و Statement - 2 يمكن أن تكون تعليمة واحدة من لغة كوبول أو مجموعة تعليمات .

مثال :

```
IF      STD - CLG NOT = PRVUS - CLG
            PERFORM PRINT - YEAR THRU EXIT - CLG
ELSE
        PERFORM PRINT - RECORD THRU EXIT - PRIN.
```

واذا كان هناك أكثر من مقارنة فيمكن أن نأخذ الشكل التالي كمثال بلغة كوبول :

```
IF STD - CLG NOT = PRVUS - CLG
        PERFORM PRINT - YEAR THRU EXIT - CLG
```

```
ELSE
        IF   STD - YEAR NOT = PRVUS - YEAR
             PERFORM PRINT - YEAR THRU EXIT - YEAR
        ELSE
             PERFORM PRINT - RECORD.
```

ويجب ملاحظة وجود هامش في السطر الثاني لتعليمة IF وكلمة ELSE وكذلك كتابة كلمة IF وكلمة ELSE الخاصة بها على نفس المستوى العمودي لسهولة القراءة والفهم.

الدوارة :

تأخذ الشكل العام

```
PERFORM Procedure - name
    UNTIL Condition - P (    =    true)
```

ومثال على ذلك التعليمات التالية :

```
        MOVE 'NO' TO END - OF - JOB.
        PERFORM READ - RECORD
         \
          \ UNTIL END - OF - JOB = 'YES'.
           \
READ - RECORD.
        READ STD - FILE
         \
          \ AT END MOVE 'YES' TO END - OF - JOB.
           \
NEXT - PARAGRAPH.
         \
          \
```

٩ ـ ٤ ارشادات في ترميز برنامج كوبول على ضوء البرمجة الهيكلية

الى جانب استخدام اسلوب البرمجة الهيكلية في وضع المخطط المنطقي لهيكل البرنامج يجب على المبرمج بلغة كوبول تتبع الارشادات التي تساعده في الوصول الى برنامج سهل وسليم المنطق وعلى الرغم من أنها لا تعتبر ضمن قواعد لغة كوبول. ولكنها تكتسب بالخبرة العملية والممارسة الطويلة مع هذه اللغة من البرمجة وقد تختلف من مركز حاسب الى آخر في تطبيقها.

قسم البيانات : DATA DIVISION

١ ـ ابتداء جميع عبارات PICTURE من نفس العمود وقد يكون عادة العمود ٤٠ تقريباً أو حوله. وعلى المبرمج توحيد الكلمة في البرنامج المستخدم وهي إما PIC أو PIC IS أو PICTURE وهذا ينطبق على عبارة VALUE وعبارة USAGE.

٢ ـ استخدام رمزين من أول الحقول الموصوفة في السجل الواحد للملف الواحد لاجل تمييزها بسهولة. مثل

ST - NAME, ST - ADDRESS, ST - TELEPHONE

٣ ـ حذف ادخالات المستوى رقم 77 ومحاولة تجميعها حسب نوعيتها ضمن المستوى رقم ∅1.

٤ ـ ترك هوامش متسلسلة في أرقام المستوى عند توصيف السجل في الملف. كذلك استخدام أرقام مستوى بينها فجوات رقمية لأجل السماح بالاضافة بالمستقبل

مثل ∅5, 1∅, 15, 2∅,

أو ∅3, ∅7, ∅9, 11, 15

٥ ـ استخدام أسماء حقول ذات دلالة ومعنى وليس مجرد حروف ورموز مبهمة مثل ST - CODE بدلاً من ST - X

وكذلك يجب التنويه الى عدم الافراط في الايضاح والتركيز على اختصار الاسم مع المحافظة على المعنى والدلالة .

٦ - استخدام هامش عند كتابة السطر الواحد على سطرين لعدم كفاية السطر الأول ، من أجل الإشارة الى استمرارية السطر الأول مثل :

```
ϕ5    FILLER    PIC X (13) VALUE
              ' ----- BOOK -----'
```

والشكل التالي يلخص ما سبق التنويه إليه :

```
ϕ1    STUDENT - IN - REC.
   ϕ5    ST - NAME.
      1ϕ    ST - FIRST - NAME               PIC  X (25).
      1ϕ    ST - LAST - NAME                PIC  X (25).
   ϕ5    ST - BIRTH - DATE.
      1ϕ    ST - BIRTH - DAY                PIC    99.
      1ϕ    ST - BIRTH - MONTH              PIC    99.
      1ϕ    ST - BIRTH - YEAR               PIC    99.
```



```
WORKING - STORAGE SECTION.
ϕ1    FLAGS
   ϕ5    TOTAL - FLAG                        PIC    9.
   ϕ5    AGE - FLAG                          PIC    9.
```

قسم الاجراءات PROCEDURE DIVISION

١ - تجنب كتابة أكثر من تعليمة على نفس السطر، وذلك من أجل سهولة المراجعة والقراءة رغم أن مترجم كوبول يقبل أكثر من تعليمة على نفس السطر.

٢ - يفضل وضع أسماء الفصول والفقرات على أسطر مستقلة، وكذلك محاولة ترك سطر فارغ قبل هذه الأسماء لزيادة الوضوح والقراءة السهلة.

٣ - إذا كان هناك أسماء فقرات أو فصول في قسم الاجراءات متسلسلة فيجب أن تكون أرقامها متسلسلة أيضاً وتحتوي على فجوات رقمية لامكانية حشر فقرات أخرى عند التعديل إذا اضطر لذلك المبرمج. مثل

```
ØØ1Ø - HOUSEKEEPING.
        \
        \
ØØ2Ø - READ - REC.
        \
        \
```

٤ - ترك فراغات مناسبة لإبراز التعليمات المهمة في البرنامج وكذلك استخدام الهوامش لهذا الغرض عند وجود تعليمة IF المتداخلة.

٥ - ربط تعليمة IF مع كلمة ELSE الخاصة بها و وضعها في نفس المستوى عمودياً عند الكتابة. لأن كل ELSE ترجع الى أقرب IF في حال وجود IF المتداخلة.

٦ - تنظيم أسماء الحقول في التعليمات المتشابهة عمودياً مثل تعليمة MOVE كما في المثال التالي :

```
MOVE    ST - NAME       TO P - NAME.
MOVE    ST - ADRESS     TO P - ADRESS.
MOVE    ST - SCORES     TO P - SCORES.
```

٧ - صياغة التعليمات بطريقة سهلة دون تعقيد خاصة في التعليمات الحسابية لئلا يسبب التحوير في العمليات الحسابية صعوبة في فهم الترميز بلغة كوبول وعلى الأخص عند استخدام تعليمة COMPUTE.

وإلى جانب هذه الارشادات في القسمين يوجد بعض النقاط المشتركة بينهما يمكن ايجازها فيما يلي :

قسم البيانات وقسم الاجراءات :

١- يفضل استخدام أسطر فارغة أثناء الترجمة وعند الكتابة فمثلا الرمز / يقوم بقلب الصفحة عند الترجمة عند وضعه في العمود ٧ أو استخدام أوامر SKIP EJECT * وترك أسطر فارغة بين التعليمات لابراز المهم منها.

٢ - الاستفادة من الأعمدة ٧٣ الى ٨٠ في صفحة الكتابة بلغة كوبول لغرض الاشارة الى التعديلات الواردة على البرنامج بعد كتابته فمثلا لو تم وضع :
BKH 05/82

للدلالة على أوائل أحرف من اسم المبرمج الذي قام بالتعديلات وهي BKH و 05 تدل على الشهر في سنة ١٩٨٢ وهو التاريخ الذي تم تعديل هذا السطر في البرنامج.

٣ - استخدام الملاحظات المناسبة بين أسطر البرنامج للإشارة الى وظيفة بعض الأجزاء أو الفقرات فيه ، مع الأخذ بعين الاعتبار عدم الافراط في ذلك.

٤ - تجنب استخدام الفواصل Commas و يكفي استخدام أعمدة فارغة بين الجمل أو العبارات التي لا تحتاج الى وجود نقطة في نهايتها ، وذلك لكي لا يختلط شكل الفاصلة مع النقطة على المبرمج.

٩ - ٥ - مثال عملي :

في الصفحات التالية نموذج عن برنامج بلغة كوبول قد صمم ورمز مع مراعاة البرمجة الهيكلية.

(*) موجودة فقط في أنظمة تشغيل IBM

```
*
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT21.
 AUTHOR.      KHAYAT-JNAID.
*
*    يقوم هذا البرنامج بطباعة المجاميع
*             الجزئية والمتوسطة والنهائية
*
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.    IBM-370.
   OBJECT-COMPUTER.    IBM-370.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
       SELECT STDFILE ASSIGN  TO UT-S-STDF.
       SELECT PRINTF  ASSIGN  TO UT-S-PRFL.
 DATA DIVISION.
 FILE SECTION.
 FD     STDFILE   LABEL RECORD  IS STANDARD
                       DATA RECORD IS STD-REC.
 01     STD-REC.
        03    STD-NO     PIC  9(5).
        03    STD-NAME   PIC  X(30).
        03    STD-CLG    PIC  99.
        03    STD-YEAR   PIC  99.
        03    STD-DATA   PIC  X(11).
*
 FD     PRINTF     LABEL RECORD  IS OMITTED
                       DATA RECORD IS  P-REC.
 01     P-REC           PIC     X(132).
*
 WORKING-STORAGE SECTION.
 01     HDR-1.
  03    FILLER  PIC    X(52)    VALUE SPACES.
  03    FILLER  PIC    X(28)    VALUE
   'KING SAUD UNIVERSITY '.
  03    FILLER  PIC    X(52)    VALUE SPACES.
 01     HDR-2.
  03    FILLER  PIC    X(11)    VALUE 'DATE:'.
  03    DAY-P   PIC    99.
  03    FILLER  PIC    X        VALUE '/'.
```

```
 03    MONTH-P PIC     99.
 03    FILLER  PIC     XXX      VALUE '/19'.
 03    YEAR-P  PIC     99.
 03    FILLER  PIC     X(31)    VALUE SPACES.
 03    FILLER  PIC     X(15)    VALUE
  'STUDENTS REPORT'.
 03    FILLER  PIC     X(34)    VALUE SPACES.
 03    FILLER  PIC     X(06)    VALUE 'PAGE :'.
 03    P-CTR   PIC     ZZ9.
 03    FILLER  PIC     X(11)     VALUE SPACES.
01     HDR-3.
 03    FILLER  PIC     X(10)    VALUE SPACES.
 03    FILLER  PIC     X(17)    VALUE
      'COLLEGE CODE'.
 03    FILLER  PIC     X(19)    VALUE
       'YEAR NO       '.
 03    FILLER  PIC     X(16)    VALUE
       'STUDENT NUMBER'.
 03    FILLER  PIC     X(16)    VALUE SPACES.
 03    FILLER  PIC     X(14)    VALUE
       'STUDENT NAME  '.

*
 01    DETAILS.
 03    FILLER  PIC     X(16)    VALUE SPACES.
 03    P-CLG   PIC     99.
 03    FILLER  PIC     X(12)    VALUE SPACES.
 03    P-YEAR  PIC     99.
 03    FILLER  PIC     X(18)    VALUE SPACES.
 03    P-NO    PIC     Z(3)99.
 03    FILLER  PIC     X(17)    VALUE SPACES.
 03    P-NAME  PIC     X(30).
 03    FILLER  PIC     X(15)    VALUE SPACES.

*
 01    TOTL-YEAR.
 03    FILLER  PIC     X(23)    VALUE
  '      TOTAL OF YEAR NO. '.
 03    P-TYEAR PIC     99.
 03    FILLER  PIC     X(15)    VALUE ':'.
 03    P-CYEAR PIC     ZZ99.
 03    FILLER  PIC     X(90)    VALUE SPACES.
```

```
*
 01     TOTL-CLG.
  03    FILLER  PIC     X(29)    VALUE
   '    ** TOTAL OF COLLEGE CODE '.
  03    P-TCLG  PIC     99.
  03    FILLER  PIC     X(5)    VALUE ':'.
  03    P-CCLG  PIC     ZZZ99.
  03    FILLER  PIC     X(90)    VALUE SPACES.
*
 01     TOTL-OV-ALL.
  03    FILLER  PIC     X(24)    VALUE SPACES.
  03    FILLER  PIC     X(32)    VALUE
   '      TOTAL OF ALL THE STUDENTS :'.
  03    P-OVAL  PIC     ZZZ999.
  03    FILLER  PIC     X(60)    VALUE SPACES.
*  لقبول التاريخ من الحاسب
 01     CURENT-DATE.
  03    C-YEAR  PIC     99.
  03    C-MONTH PIC     99.
  03    C-DAY   PIC     99.
*      عدادات تستخدم للتجميع
 01   TOTAL-COUNTERS.
  03 YEAR-CNTR PIC     9(4).
  03 COLG-CNTR PIC     9(4).
  03 OVAL-CNTR PIC     9(6).
* عدادات تستخدم للصفحة والسطر
 01    LINE-PAGE-COUNTERS.
  03 LN-CNTR    PIC     99.
  03 PG-CNTR    PIC     999.
* مساحات عمل تستخدم لمقارنة السجل الحالى *
*       مع السابق بعد ان نكون قد حفظنا فيها
* القيم السابقة للحقول المعنية بالمعالجة *
 01   OLD-VALUES.
  03 PRVS-YEAR PIC     99.
  03 PRVS-CLG  PIC     99.
*
 77   END-OF-JB PIC     XXX.
```

```
*
 PROCEDURE DIVISION.
       OPEN       INPUT    STDFILE
                   OUTPUT   PRINTF.
* قبول التاريخ من الحاسب وطباعة *
*                             العناوين
       ACCEPT    CURENT-DATE FROM DATE.
       MOVE      C-YEAR          TO    YEAR-P.
       MOVE      C-MONTH         TO    MONTH-P.
       MOVE      C-DAY           TO    DAY-P.
       PERFORM   NEW-PAGE        THRU EXTP.
* التعليمات الخمس التالية تشكل المعالجة *
*                          الرئيسة في البرنامج
       PERFORM  READ-FILE  THRU      EXIT-READ.
       MOVE   STD-CLG   TO  PRVS-CLG  P-TYEAR.
       MOVE   STD-YEAR TO  PRVS-YEAR P-TCLG.
       PERFORM  MAIN-LOOP  THRU      EXIT-LOOP
                UNTIL      END-OF-JB = 'YES'.
     PERFORM    END-JOB    THRU      EXIT-END.
     STOP       RUN.

*
 MAIN-LOOP.
     IF     STD-CLG   NOT EQUAL PRVS-CLG
            PERFORM  PRINT-YEAR THRU  EXIT-CLG
     ELSE
     IF     STD-YEAR  NOT EQUAL PRVS-YEAR
        PERFORM  PRINT-YEAR THRU  EXIT-YEAR
     ELSE
        PERFORM   READ-FILE  THRU  EXIT-READ.
 EXIT-LOOP.
       EXIT.

*
*         قراءة سجل من الملف لمعالجته
*
 READ-FILE.
        READ STDFILE AT END
               MOVE 'YES' TO END-OF-JB.
 EXIT-READ.
         EXIT.
```

```
*
*    :    طباعة سجل دارس
*
 PRINT-RECORD.
       MOVE    STD-NO     TO        P-NO.
       MOVE    STD-NAME   TO        P-NAME.
       MOVE    STD-CLG    TO        P-CLG.
       MOVE    STD-YEAR   TO        P-YEAR.
       WRITE   P-REC      FROM      DETAILS
                          AFTER     2.
       ADD     1          TO        LN-CNTR.
       IF      LN-CNTR    >         18
               PERFORM    NEW-PAGE  THRU  EXTP.
       ADD     1          TO        YEAR-CNTR.
 EXIT-PRINT.
       EXIT.

*
*    :    طباعة مجاميع الطلاب لسنة معينة
*
 PRINT-YEAR.
       MOVE     YEAR-CNTR  TO       P-CYEAR.
       WRITE    P-REC      FROM     TOTL-YEAR
                           AFTER    2.
       ADD      YEAR-CNTR  TO       COLG-CNTR.
       MOVE     STD-YEAR   TO       PRVS-YEAR
                                    P-TYEAR.
 EXIT-YEAR.
       EXIT.

*
*         طباعة مجاميع الطلاب لكلية معينة
*
 PRINT-CLG.
       MOVE     COLG-CNTR  TO       P-CCLG.
       WRITE    P-REC      FROM     TOTL-CLG
                           AFTER    2.
       ADD      COLG-CNTR  TO       OVAL-CNTR.
       MOVE     STD-CLG    TO       PRVS-CLG
                                    P-TCLG.
 EXIT-CLG.
       EXIT.
```

```
*
*    فقرة قلب الصفحة وكتابة العناوين:                          *
*
 NEW-PAGE.
        MOVE      ZERO          TO       LN-CNTR.
        ADD       1             TO       PG-CNTR.
        MOVE      PG-CNTR       TO       P-CTR.
        MOVE      SPACES        TO       P-REC.
        WRITE     P-REC         AFTER    PAGE.
        WRITE     P-REC         FROM     HDR-1
                                AFTER    1.
        WRITE     P-REC         FROM     HDR-2
                                AFTER    3.
        WRITE     P-REC         FROM     HDR-3
                                AFTER    2.
  EXTP.
        EXIT.
*
*    فقرة طباعة المجاميع النهائية                              *
*
 END-JOB.
        PERFORM PRINT-YEAR   THRU EXIT-CLG.
        MOVE      OVAL-CNTR      TO    P-OVAL.
        WRITE     P-REC         FROM TOTL-OV-ALL
                                   AFTER 2.
        CLOSE       STDFILE
                    PRINTF.
 EXIT-END.
        EXIT.
```

٩ ـ ٦ اسئلة وتمارين

سؤال ١ - إذا كان لدينا قسم الاجراءات التالي :

```
PROCEDURE DIVISION.
MAIN - P SECTION.
        PERFORM FIRST - RTN.
        PERFORM SECOND - RTN.
        PERFORM THIRD - RTN UNTIL TOT - SUM      12.
        STOP RUN.
NEXT - P SECTION.
FIRST - RTN.
        MOVE 1 TO N.
        MOVE ZEROS TO TOT - SUM.
SECOND - RTN.
        COMPUTE     M = N   *   2.
        COMPUTE     X = N   *   3.
        ADD 1 TO N.
        ADD M, X, N, GIVING SUM - 1.
THIRD - RTN.
        ADD SUM - 1 TO TOT - SUM.
FOURTH - RTN.
        ADD 1 TO N.
        ADD  2 TO X.
```

NEXT - Q SECTION.

.....

المطلوب :

أ ـ ما هي القيمة النهائية للحقل SUM - 1؟

ب ـ ما هي القيمة النهائية للحقل TOT - SUM ؟

جـ ـ كم مرة تُنفذْ الفقرة THIRD - RTN ؟

د ـ هل هناك فقرات لا تنفذ وما هي؟

هـ ـ هل هناك فقرات تُنفذ أكثر من مرة وما هي ؟

سؤال ـ ٢ في السطور التالية برنامج مكتوب بلغة كوبول اسمه NDXFILE يقوم بعملية إنشاء ملف مفهرس INDEXED من بطاقات مثقبة لبيانات عن الدارسين. والمطلوب تتبع هذا البرنامج وملء الفراغات بالشكل الصحيح.

```
000110   IDENTIFICATION DIVISION.
000120   PROGRAM - ID. NDXFILE.
000130   AUTHOR. AUTHORS.
000140**                THIS PROGRAM IS TO DEMONSTRATE.
000150**                INDEXED FILE CREATION.
000160   ENVIRONMENT DIVISION.
000170   CONFIGURATION SECTION.
000180   SOURCE - COMPUTER. IBM - 3033.
000190   OBJECT - COMPUTER. IBM - 3033.
000191   INPUT - OUTPUT SECTION.
000192   ............ ◄──────────────────────────── (1)
```

```
000193        SELECT CARD - FILE ASSIGN TO UT - S - SYSIN.
000194        SELECT DISK - FILE ASSIGN TO VSAMFILE
000195           ORGANIZATION IS INDEXED
000196           ACCESS MODE IS ...... ←——————————(2)
000197           RECORD KEY IS STUDENT - NUMBER.
000198        SELECT PRINT - FILE ASSIGN TO UT - S - PRF.
000199    DATA DIVISION.
000200    FILE SECTION.
000201    FD DISK - FILE LABEL RECORD IS ........ ←——————(3)
000202              RECORD CONTAINS 80 CHARACTERS
000203              DATA RECORD IS DISK - REC.
000204    01        DISK - REC.
000205        05    STUDENT - NUMBER     PIC ..... ←———————(4)
000206        05    STUDENT - DATA      PIC X (60).
000207        05    FILLER      PIC X (12).
000208    FD        CARD - FILE LABEL RECORD IS ....... ←————(5)
000209              DATA RECORD IS CARD - REC.
000210    01        CARD - REC.
000211        05    STUDENT - IDENT          PIC 9 (8).
000212        05    STUDENT - DATA           PIC X (60).
000213        05    FILLER               PIC X (12).
000215    FD        PRINT - FILE LABEL RECORD IS OMITTED
000216              DATA RECORD IS PRINT - REC.
000217    01        PRINT - REC          PIC X (132).
000218*
```

```
000219  WORKING - STORAGE SECTION.
000220  01        END - OF - DATA PIC X VALUE 'N'.
000221      88        INPUT - END VALUE 'Y'.
000222*
000223  PROCEDURE DIVISION.
000224  MAINROUTINE.
000225      OPEN INPUT CARD - FILE  <------------------------ (6)
000226          ...... PRINT - FILE, DISK - FILE.
000227      PERFORM READ - CARD.
000228      PERFORM FILE - CREATE UNTIL INPUT - END.
000229      CLOSE CARD - FILE, ......... , PRINT - FILE.
                                 <------------------------ (7)
000230       STOP RUN.
000231*
000249  FILE - CREATE.
000250       MOVE STUDENT - IDENT TO STUDENT - NUMBER.
000260       MOVE STUDENT - DATA OF ........ TO  <----------- (8)
000270               STUDENT - DATA OF .........
                                      <------------------- (9)
000280       WRITE DISK - REC
000290           INVALID KEY ........ ERROR - CASE.
                              <--------------------------- (10)
000300       PERFORM READ - CARD.
000400  READ - CARD.
000500       READ CARD - FILE RECORD  <------------------- (11)
000600                 AT END MOVE ..... TO END - OF - DATA.
000700  ERROR - CASE.
000800        MOVE 'RECORD EXIST IN THIS FILE'
```

```
000900              TO PRINT - REC.
001000          WRITE PRINT - REC AFTER ADVANCING 1 LINE.        (12)
001100          WRITE ........ FROM CARD - REC AFTER ADVANCING 1 LINE.
001200      EXIT - PAR. EXIT.
```

الفصل العاشر

تدقيق البرنامج واختباره

١٠ ـ ١ لمحة عامة

من المتوقع جداً وجود أخطاء في برنامج كوبول خاصة تلك البرامج التي تكتب في المراحل الأولى من تعلم هذه اللغة . ويمكن تصنيف الأخطاء الى نوعين :

١ ـ أخطاء شكلية وتتعلق بقواعد اللغة واستخدام الكلمات الخاصة بها . وتظهر هذه الأخطاء أثناء عملية الترجمة ERRORS in COMPILATION .

٢ ـ أخطاء منطقية وتظهر هذه الأخطاء بعد خلو البرنامج من الأخطاء الشكلية أولاً ، ويكتشفها المبرمج بعد وضع البرنامج موضع التنفيذ (أو الاختبار) . ويمكن حصر هذه الأخطاء في حالتين :

أ ـ امكانية تنفيذ البرنامج ولكن يتم الحصول على نتائج غير سليمة .
ب ـ توقف تنفيذ البرنامج بشكل غير طبيعي عند بعض فقرات أو تعليمات البرنامج .

١٠ ـ ٢ الأخطاء الشكلية ERRORS IN COMPILATION

إن كل مترجم كوبول يقوم بعملية فحص وتدقيق شكل السطر أو التعليمة المكتوبة وفق القواعد والأسس العامة لهذه اللغة وعند اكتشافه لأي عيب أو خطأ في هذه القواعد يعط إشارة خطأ على هذا السطر واحياناً مع الاشارة الى مكان الخطأ في

نفس السطر* وتصنف هذه الأخطاء الى نوعين بشكل عام :

- أخطاء أساسية و يرمز لها عادة بالحرف E ولا يمكن متابعة تنفيذ البرنامج بدون تصحيحها.
- أخطاء مميتة** و يرمز لها عادة بالحرف D وعند ظهورها في البرنامج يتوقف المترجم عن العمل بعدها.
- أخطاء تحذيرية و يرمز لها بالحرف W وفي بعض الأجهزة بالحرف C، حيث يقبلها المترجم ولكن يعطي إشارة تحذيرية للمبرمج على مكان وجودها. ويمكن متابعة تنفيذ البرنامج بدون تصحيحها.
- أخطاء شرطية*** و يرمز لها بالرمز C و يفترض المترجم الحالة الصحيحة منها ويمكن للمبرمج متابعة تنفيذ البرنامج بدون تعديلها.

ويجب لفت نظر المبرمج الى أن المترجم قد يزيد الطين بلة عند اكتشافه لخطأ شكلي في أول البرنامج حيث أنه يعطي نفس الإشارة للدلالة على الخطأ برقم جديد عند كل استخدام لهذا الحقل أو الملف أو السجل في الأقسام التالية من البرنامج، وأحياناً يكون عدد الأخطاء الشكلية Syntax بالمئات في نهاية عملية الترجمة إلا أن السبب أو الخطأ الرئيسي يكون قليلاً جداً ولكن موقع هذا الخطأ يكون في مستهل البرنامج.

١٠ - ٣ الأخطاء المنطقية ERRORS IN EXECUTION

هذه الأخطاء تظهر عند اختبار البرنامج و بعد خلوه من أخطاء الشكل أو القواعد. وعند تدقيق النتائج الناجمة من عملية الاختبار يكتشف المبرمج هذه الأخطاء، وهناك

(*) تختلف باختلاف المترجم المستخدم

(**) Crawford and Grauer P . 72

(***) Conditional ويمكن أن تصنف مع الأخطاء التحذيرية في بعض المترجمات غير الخاصة بشركة IBM.

حالتان لظهور هذه الأخطاء :

أ ـ انتهاء البرنامج المنفذ بشكل طبيعي مع الحصول على نتائج غير سليمة . وهذه الأخطاء في النتائج يمكن تفسيرها على شكلين .

ـ إما خطأ في سير منطق العمليات ضمن البرنامج و بشكل مخالف للمخطط الانسيابي Flowchart الموضوع لهذا البرنامج .

ـ أو خطأ في تطابق البيانات الداخلة الى البرنامج مع الشكل Format الموضوع لوصف هذه البيانات المدخلة .

و بالتالي يجب على المبرمج التأكد من تطابق وصحة البيانات المقروءة الى البرنامج من الملفات المخصصة لإدخال المعلومات ومعالجتها .

ب ـ توقف البرنامج عن التنفيذ وانتهائه بشكل غير طبيعي أو دخوله في دوارة لا نهاية لها Endless Loop . وهذا ينجم عن خطأ منطقي في نقل المخطط الانسيابي الى رموز البرنامج أو أحياناً عن استخدام غير سليم للتعليمات الموضوعة للغة كوبول .

١٠ ـ ٤ أخطاء شائعة

هناك أخطاء شائعة الحدوث في لغة كوبول يمكن تلخيصها في النقاط التالية لتساعد المبرمج في تحديد الخطأ الحاصل في البرنامج :

١ ـ اغفال تحديد نقطة البداية* في الحقل أو المؤشر Subscript .

٢ ـ وضع بيانات غير رقمية في حقول موصوفة رقمياً .

٣ ـ الاستخدام غير الصحيح للنقطة في نهاية الجملة بلغة كوبول .

٤ ـ خروج قيمة المؤشر أو الدليل عن مجال الجدول الموضوع لأجله .

(*) Initialization

٥ - التداخل غير الصحيح في تعليمة PERFORM .

٦ - معالجة سجل ادخال مرتين .

٧ - عدم تطابق صيغة البيانات في الحجم والشكل الموصوفين في السجل Record .

١٠ - ٥ كشف الاخطاء وتصحيحها

يجب على المبرمج قبل الوقوع في الأخطاء من مختلف الأنواع مراعاة النقاط التالية أثناء تصميم برامجه للحصول على أدنى نسبة من الخطأ :

- تجنب التعليمات Instructions غير الواضحة والمبهمة مثل تعليمة المقارنة IF مع العوامل المنطقية NOT, OR , AND .

- الاستفادة قدر الامكان من اسلوب البرمجة الهيكلية Structured Programming (الفصل رقم ٩) وكذلك استخدام اسلوب التصميم المجزأ Modular Programming من أعلى الى اسفل عند وضع الخطوات الأساسية للبرنامج .

- اتباع اسلوب برمجة سهل وخالي من التعقيد و بالتالي خالي من الأخطاء .

وفي حال حدوث الأخطاء يمكن للمبرمج اللجوء الى الطرق التالية من أجل اكتشافها و بالتالي تصحيحها ثم اعادة اختبار البرنامج .

- تدقيق البرنامج من قبل المبرمج نفسه للكشف عن الخطأ ، والتحري عن الخطأ البسيط والأكثر احتمالا ثم البحث عن الخطأ المعقد . و يستعان باستخدام الشك (الظن) في مواقع الخطأ المحتملة ، ووضع نقاط علاّمة في البرنامج .

- الاستعانة بميزة أو خاصية كشف الخطأ في لغة كوبول COBOL DEBUGGING وهذه الميزة تعطي المبرمج امكانية متابعة البرنامج

خطوة خطوة أثناء تنفيذه و بالتالي معرفة مواقع الخطأ*.

- استخدام أوامر نضح (تفريغ) للبرنامج أثناء التنفيذ و يدعى الأمر DUMP وبموجبه يمكن للمبرمج من تفريغ محتو يات الذاكرة أثناء تنفيذ البرنامج، وعند توقف البرنامج غير الطبيعي أو عند حصول الخطأ الذي صعب كشفه بالطرق السابقة، وغالباً ما تكون أشكال DUMP مكتوبة بالرموز السادسية عشر Hexadecimal وتبين فيها عناو ين تخزين فقرات البرنامج، وتختلف من حاسب لآخر لذلك ينصح بالرجوع الى مراجع الشركة المختصة، لمعرفة امكانية قراءة هذه الكشوف الناجمة عن عملية التفريغ.

١٠ - ٦ تمارين

١ - في الصفحات التالية برنامج كو بول و يظهر بعد ترجمته بوساطة مترجم نظام IBM وتبدو الأخطاء الناجمة عن عملية الترجمة Compilation في الصفحة الأخيرة من البرنامج وقد تم ذكر رقم السطر الذي يحتوي هذا الخطأ والمطلوب كشف الأخطاء وتصحيحها ليصبح البرنامج خالياً من أي منها.

(*) تكون متوفرة في بعض المترجمات الخاصة بلغة كو بول. وللمزيد من المعلومات ينصح بالرجوع الى مراجع الشركة. أو الى الكتب المتقدمة في لغة كو بول.

```
001  *
002   IDENTIFICATION DIVISION.
003   PROGRAM-ID. TESTP.
004   AUTHOR.      KHAYAT-JNAID.
005  *
006  *
007  *
008  *
009  *
010   ENVIRONMENT DIVISION.
011   CONFIGURATION SECTION.
012   SOURCE-COMPUTER.   IBM-370.
013   OBJECT-COMPUTER.   IBM-370.
014   INPUT-OUTPUT SECTION.
015   FILE CONTROL.
016        SELECT STDCKF ASSIGN  TO UT-S-INF1.
017        SELECT SRTFIL ASSIGN  TO UT-S-INF3.
018        SELECT OUTSRT ASSIGN  TO UT-S-MRGF.
019        SELECT PRINTF ASSIGN  TO UT-S-PRFL.
020   DATA DIVISION.
021   FILE  SECTION.
022  *
023   FD    STDCKF LABEL RECORD IS STANDARD
024              RECORD  CONTAINS 80 CHARACTERS
025                      DATA RECORD IS STK-REC.
026   01    STK-REC         PIC  X(80).
027  *
028   SD    SRTFIL
029               RECORD  CONTAINS 80 CHARACTERS
030                         DATA RECORD IS SRT-REC.
031  *
032  *
033   01    SRT-REC.
034    03 SRT-NO        PIC    9(6).
035    03 SRT-DATA      PIC    X(74).
036  *
037   FD    OUTSRT LABEL RECORD  STANDARD
038             RECORD  CONTAINS 80 CHARACTERS
039                         DATA RECORD IS OUT-REC.
040   01    OUT-REC.
041     03 OUT-NO         PIC    9(5).
```

```
042      03 OUT-DESC      PIC     X(20).
043      03 OUT-QTY       PIC     9(5).
044      03 OUT-CODE      PIC     9(3).
045      03 OUT-PRC       PIC     9(3)V99.
046      03 OUT-MIN       PIC     9(5).
047      03 FILLER        PIC     X(37).
048   *
049    FD    PRINTF     LABEL RECORD  IS OMITTED
050                        DATA RECORD IS  P-REC.
051    01    P-REC           PIC      X(132).
052   *
053   *
054    WORKING-STORAGE SECTION.
055    01    HDR1.
056     03   FILLER  PIC     X(57)    VALUE SPACES.
057     03   FILLER  PIC     X(17)    VALUE
058                         'S T O C K  L I S T'.
059     03   FILLER  PIC     X(56)    VALUE SPACES.
060   *
061    01    HDR2.
062     03   FILLER  PIC     X(08)    VALUE
063                                      ' DATE :'.
064     03   P-DATE  PIC     99/99/99.
065     03   FILLER  PIC     X(100)  VALUE SPACES.
066     03   FILLER  PIC     X(07)    VALUE
067                                      'PAGE :'.
068     03   P-PAGE  PIC     ZZ9.
069     03   FILLER  PIC     X(5)     VALUE SPACES.
070   *
071    01    HDR3.
072     03   FILLER PIC      X(40)    VALUE
073          '     ITEM NO.     ITEM DESCRIPTION'.
074     03   FILLER PIC      X(29)    VALUE
075               '      QUANTITY     ITEM CODE'.
076     03   FILLER PIC      X(17)    VALUE
077                              '    ITEM PRICE'.
078     03   FILLER PIC      X(46)    VALUE
079         '     MINIMUM LEVLEL        COMMENTS'.
080   *
081    01    DATA-LINE.
082     03   FILLER  PIC     X(06)    VALUE SPACES.
```

```
083       03    P-NO     PIC     ZZZZ99.
084       03    FILLER   PIC     X(06)    VALUE SPACES.
085       03    P-DESC   PIC     X(20).
086       03    FILLER   PIC     X(09)    VALUE SPACES.
087       03    P-QNTY   PIC     ZZZZ9.
088       03    FILLER   PIC     X(09)    VALUE SPACES.
089       03    P-CODE   PIC     Z99.
090       03    FILLER   PIC     X(10)    VALUE SPACES.
091       03    P-PRCE   PIC     ZZ9.99.
092       03    FILLER   PIC     X(10)    VALUE SPACES.
093       03    P-MINI   PIC     Z(4)9.
094       03    FILLER   PIC     X(12)    VALUE SPACES.
095       03    P-COMNT  PIC     X(7).
096       03    FILLER   PIC     X(10)    VALUE SPACES.
097   *
098   *
099   *
100    01    C-DATE   PIC     9(6).
101    01    PAG-CTR  PIC     999      VALUE ZERO.
102    01    LIN-CTR  PIC     99       VALUE 25.
103    77    FLAG     PIC     9        VALUE ZERO.
104  *
105    PROCEDURE DIVISION.
106  *
107  *
108  *
109          SORT   SRTFIL
110                   ASCENDING KEY SRT-NO
111            USING  STOCKF
112                   GIVING    OUTSRT.
113  *
114  *
115  *
116  *
117   START-PROCESSING.
118       OPEN INPUT OUTSRT
119            OUTPUT   PRINTF.
120        ACCEPT C-DATE FROM DATE.
121        MOVE C-DATE TO P-DATE.
122          READ OUTSRT AT END
123                MOVE  1   TO   FLAG.
124  *
```

```
125   MAIN-LOOP.
126         PERFORM MOVE-WRITE
127                     THRU EXT-MOVE-WRITE
128          UNTILE FLAG EQUAL 1.
129          CLOSE OUTSRT PRINTF.
130          STOP RUN.
131  *
132   MOVE-WRITE.
133          IF      OUT-QTY   NOT    > TOT-MIN
134          MOVE    'REORDER'  TO     P-COMNT
135          ELSE
136          MOVE     SPACES    TO     P-COMNT.
137          MOVE    OUT-NO     TO     P-NO.
138          MOVE    OUT-DESC   TO     P-DESC.
139          MOVE    OUT-QTY    TO     P-QTY.
140          MOVE    OUT-CODE   TO     P-CODE.
141          MOVE    OUT-PRC    TO     P-PRCE.
142          MOVE    OUT-MIN    TO     P-MINI.
143          IF      LIN-CTR    >      24
144                  PERFORM    NEW-PAGE.
145          WRITE  P-REC  FROM  DATA-LINE.
146          ADD    1      TO    LIN-CTR.
147          READ OUTSRT   AT END
148                    MOVE  1   TO   FLAG.
149     EXT-MOVE-WRITE.
150          EXIT.
151     NEW-PAGE.
152         MOVE  SPACES   TO       P-REC.
153         WRITE P-REC    AFTER    PAGE.
154         MOVE  ZERO     TO       LIN-CTR.
155         ADD   1        TO       PAG-CNTR.
156         MOVE  PAG-CNTR TO       P-PAG.
157         WRITE P-REC    FROM     HDR1
158                            AFTER 1 LINE.
159         WRITE P-REC    FROM     HDR2
160                        AFTER 1 LINE.
161         WRITE P-REC    FROM     HDR3
162                             AFTER 2 LINE.
163     EXTP.
164         EXIT.
165         *
```

```
00165         *
_ CARD   ERROR MESSAGE

 15      IKF1087I-W     ' FILE ' SHOULD NOT BEGIN A-MARGIN.
 15      IKF1004I-E     INVALID WORD FILE . SKIPPING TO NEXT RECOGNIZABLE WORD.
 16      IKF1003I-W     FILE-CONTROL PARAGRAPH NAME MISSING. ASSUMED PRESENT.
 57      IKF2126I-C     VALUE CLAUSE LITERAL TOO LONG. TRUNCATED TO PICTURE SIZE
 111     IKF1161I-E     STOCKF  IS NOT A FILE-NAME IN USING CLAUSE.
 109     IKF4155I-E     SYNTAX REQUIRES 1 TO 16 FILE-NAMES IN SORT USING CLAUSE.
                        DISCARDED.
 126     IKF3001I-E     UNTILE NOT DEFINED. DELETING TILL LEGAL ELEMENT FOUND.
 133     IKF3001I-E     TOT-MIN NOT DEFINED. TEST DISCARDED.
 139     IKF3001I-E     P-QTY NOT DEFINED. DISCARDED.
 155     IKF3001I-E     PAG-CNTR NOT DEFINED. SUBSTITUTING TALLY .
 156     IKF3001I-E     PAG-CNTR NOT DEFINED. DISCARDED.
 156     IKF3001I-E     P-PAG NOT DEFINED.
```

٢ ـ في الصفحات التالية برنامج كوبول و يظهر بعد ترجمته بوساطة مترجم TI 990 وتبدو الأخطاء الناتجة عن عملية الترجمة COMPILATION في صفحات البرنامج تحت الأسطر الحاوية على أخطاء.

والمطلوب كشف هذه الأخطاء وتصحيحها ليصبح البرنامج خالياً من الأخطاء القاعدية.

```
            *
            *   صحح الأخطاء الواردة فى هذا البرنامج
            *

 4          IDENTIFICATION DIVISION.
 5           PROGRAM-ID.     ERR33.
 6            AUTHOR.          KHAYAT-JNAID.
 7           *
 8             ENVIRONMENT     DIVISION.
 9             CONFIGURATION   SECTION.
10             SOURCE-COMPUTER. TI990.
11             OBJECT-COMPUTER. TI990.
12             FILE-CONTROL.
               $
*****   1) SYNTAX       *E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E
*****   1) SCAN RESUME  *W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W
13                 SELECT SUBSFL ASSIGN TO RANDOM "SM"
14                 ORGANIZATION  IS     INDEXED;
15                 ACCESS MODE   IS     RANDOM;
16                 RECORD KEY    IS     SUBS-NO.
17            *
18             DATA            DIVISION.
19             FILE            SECTION.
20             FD    SUBSFL    LABEL RECORD STANDARD
21                             RECORD CONTAINS 80 CHARACTERS.
22            01     SUBS-REC.
              $
*****   1) SYNTAX     *W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W
23                03 SUBS-NO          PIC    X(05).
24                03 SUBS-NAME        PIC    X(20).
25                03 SUBS-ADRES       PIC    X(24).
26                03 SUBS-DATE.
27                   05  SUBS-DAY   PIC 99.
28                   88  DAYY     VALUES ARE 01 THRU 30.
29                   05  SUBS-MONTH PIC 99.
30                   88  MONTHH   VALUES ARE 01 THRU 12.
31                   05  SUBS-YEAR  PIC 9999.
32                   88  YEARR    VALUES ARE 1401 THRU 1410.
```

```
33                     03 SUBS-CODE         PIC 99.
34                     88  CODEE        VALUES ARE 01 THRU 10.
35                     03 SUBS-AREA         PIC 99.
36                     88  AREAA        VALUES ARE 01 THRU 20.
37                     03 SUBS-CNTR         PIC 9(7).
38                     03 SUBS-LREAD        PIC 9(7).
39                     03 FILLER            PIC X(5).
40               *
41                WORKING-STORAGE       SECTION.
42               *
43                01   DEC-CODE             PIC 99
44                               USAGE IS COMP-1.
45                01   BIN-VAL REDEFINES DEC-CODE PIC XX.
46                01   CHAR                 PIC 99.
47                77   FLAG                 PIC 9.
48                77   FLD-NO               PIC 99.
49                77   DUMMY                PIC XX.
50                77   ANSR                 PIC X.
51                77   OPTIONS              PIC 9.
52               *
53                PROCEDURE          DIVISION.
54   >0000        STRT.
55   >0000              OPEN    I-O SUBSFL.
56               * التعليمات الثلاث التالية تستخدم لتحويل نمط الكتابة
57               *                             على الشاشة الى العربى
58               *
59   >0006              MOVE  18      TO DEC-CODE.
60   >000A              MOVE BIN-VAL  TO CHAR.
61   >000E              DISPLAY          CHAR.
62               *
63   >0014        MENU.
64   >0014              DISPLAY "برنامج بناء وتحديث الملف الاساسى"
65                               LINE 3 POSITION 24 ERASE
66                              "الشركة الكهرباء"
67                               LINE 4 POSITION 30
68                              "------------------------------"
69                               LINE 5 POSITION 24
70                              "اضافة سجل جديد-1"
```

```
*****    1) SYNTAX    *E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E
 71                                       LINE 6 POSITION 27
 72
 73                                      "تعديل سجل موجود2-"
 74                                       LINE 8 POSITION 27
 75                                      "حذف سجل   موجود3-"
 76                                       LINE 10 POSITION 27
 77                                      "للاستفسار عن سجل4-"
 78                                       LINE 12 POSITION 27
 79                                      "ادخل رقم العملية او اضغط RETURN للانتهاء"
 80                                       LINE 14 POSITION 20
 81                                      "-------------------------------------"
 82                                       LINE 16 POSITION 20.
                                                          $
*****    1) SCAN RESUME  *W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W
 83    >0018          ACPT-COD.
 84    >0018                ACCEPT OPTIONS
 85                                LINE 12 POSITION 70 PROMPT TAB.
 86    >0022                IF OPTIONS LESS    THAN 1 OR
 87                            OPTIONS GREATER THAN 4
 88
 89                         CLOSE SUBSFL STOP RUN.
 90               *
 91               *رمز العملية من الشاشة ومعرفة هل نحن امام اضافة ام حذف ام تعديل
 92               *
 93    >0036                MOVE HIGH VALUE TO SUBS-NO.
                                 $                $
*****    1) RESERVED WORD CONFLICT   *E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E
*****    2) SCAN RESUME  *W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W
 94    >0038                IF OPTIONS = 1 PERFORM INSRT-RTN THRU EXT-INS
                                           UNTIL SUBS-NO = LOW-VALUES.
 95
 96    >0048                IF OPTIONS = 2 PERFORM UPDAT-RTN THRU EXT-UPD
 97                                        UNTIL SUBS-NO = LOW-VALUES.
 98    >0058                IF OPTIONS = 3 PERFORM DELT-RTN  THRU EXT-DEL
 99                                        UNTIL SUBS-NO = LOW-VALUES.
100    >0068                IF OPTIONS = 4 PERFORM INQR-RTN  THRU EXT-INQ
101                                        UNTIL SUBS-NO = LOW-VALUES.
102    >0078                GO TO MENU.
103               *
```

```
104                 * الفقره الخاصه بالتعديل وفيها تتم قراءه السجل المطلوب
105                 * ثم تظهر معلوماته على الشاشه وتعدل حقوله حسب المطلوب
106                 * REWRITE          وتعاد كتابه السجل على الملف بتعليمه
107                 *
108   >007A          UPDT-RTN.
109   >007A                  PERFORM ACCEPT-NO.
110   >007C                  IF SUBS-NO = LOW-VALUES GO TO EXT-UPD.
111   >0084                  IF FLAG    = 1
112                             DISPLAY         "السجل غير موجود"
113                                             LINE 20 POSITION 10
114                                             ACCEPT  DUMMY GO TO EXT-UPD.
115   >0098                  PERFORM  DISPLAY-SCREEN.
116   >009A                  PERFORM  DISPLAY-FIELDS.
117   >009C                  DISPLAY
118                          "ادخل رقم الحقل المراد تعديله او الرقم صفر للانتهاء"
119                          LINE     23 POSITION 5 ACCEPT DUMMY.
120   >00A8                  PERFORM  ACCEPT-FLD UNTIL FLD-NO = 0.
121   >00B2                  REWRITE  SUBS-REC.
122   >00BE                  DISPLAY
123                          "السجل عدل"
124                          LINE     23 POSITION 5.
125   >00C6                  ACCEPT   DUMMY.
126   >00CA                  GO TO    UPDT-RTN.
127   >00CC          ACCEPT-FLD.
128   >00CC                  ACCEPT  FLD-NO LINE 23 POSITION 75 PROMPT TAB.
129   >00D6                  IF FLD-NO = 2    PERFORM AC-NAME     ELSE
130                          IF FLD-NO = 3    PERFORM AC-ADRES    ELSE
131                          IF FLD-NO = 5    PERFORM AC-DAY      ELSE
132                          IF FLD-NO = 6    PERFORM AC-MONTH    ELSE
133                          IF FLD-NO = 7    PERFORM AC-YEAR     ELSE
134                          IF FLD-NO = 8    PERFORM AC-CODE     ELSE
135                          IF FLD-NO = 9    PERFORM AC-AREA     ELSE
136                          IF FLD-NO = 10   PERFORM AC-CNTR     ELSE
137                          IF FLD-NO = 11   PERFORM AC-LREAD.
138   >0130          EXT-UPD.
139   >0130                  EXIT.
140                 *
141                 * الفقره التاليه هي لاضافه سجل جديد وعمل تحقق عليه
142                 *          WRITE ثم الكتابه على السجل بتعليمه
```

```
143                 *
144   >0132          INSRT-RTN.
145   >0132                PERFORM ACCEPT-NO.
146   >0134                IF SUBS-NO = LOW-VALUES GO TO EXT-INS.
147   >013C                IF FLAG    = ZERO DISPLAY
                                "السجل مكرر فى الملف الرجاء اعد العمل"
148                                 LINE 20 POSITION 10
149                                 ACCEPT DUMMY  GO TO EXT-INS.
150                        PERFORM DISPLAY-SCREEN THRU EXT-SC.
151   >0150                DISPLAY SUBS-NO LINE 7 POSITION 45.
152   >0152
153                 *  فى الفقرات التاليه يتم قبول كل حقل ويعمل تحقق عليه وتظهر
154                 *  رسالة خطا فى حل حدوثه يعاد قبوله من جديد
155                 *  *** لاحظ استخدام اسم الشرط عند التحقق من حقول التاريخ مثلا
156                 *
157   >015C          AC-NAME.
158   >015C               ACCEPT SUBS-NAME LINE 8 POSITION 45 PROMPT TAB.
159   >0166               IF     SUBS-NAME = SPACES OR SUBS-NAME NUMERIC
                                                       GO TO   AC-NAME.
160
161                 *
162   >0174          AC-ADRESS.
163   >0174               ACCEPT SUBS-ADRES LINE 9 POSITION 45 PROMPT TAB.
164   >017E               IF SUBS-ADRES =    SPACES GO TO AC-ADRESS.
165                 *
166   >0188          AC-DAY.
167   >0188               ACCEPT SUBS-DAY  LINE 11 POSITION 45 SIZE 2
                                                            PROMPT TAB.
168
169   >0194               IF NOT DAYY GO TO AC-DAY.
170                 *
171   >01A4          AC-MONTH.
172   >01A4               ACCEPT SUBS-MONTH LINE 12 POSITION 45 SIZE 2
                                                             PROMPT TAB.
173
174   >01B0               IF NOT MONTHH   GO TO AC-MONTH.
175                 *
176   >01C0          AC-YEAR.
177   >01C0               ACCEPT SUBS-YEAR LINE 13 POSITION 45 SIZE 4
                                                            PROMPT TAB.
178
179   >01CC               IF NOT YEARR   GO TO AC-YEAR.
180                 *
181   >01DC          AC-CODE.
```

```
182   >01DC                 ACCEPT SUBS-CODE LINE 14 POSITION 45 PROMPT TAB.
183   >01E6                 IF NOT CODEE GO TO AC-CODE.
184              *
185   >01F6       AC-AREA.
186   >01F6                ACCEPT SUBS-AREA LINE 15 POSITION 45 PROMPT TAB.
187   >0200                IF NOT AREAA GO TO AC-AREA.
188              *
189   >0210       AC-CNTR.
190   >0210                ACCEPT SUBS-CNTR LINE 16 POSITION 45 PROMPT TAB.
191   >021A                IF SUBS-CNTR NOT NUMERIC OR SUBS-CNTR = ZERO
192                                                    GO TO AC-CNTR.
193              *
194   >0228       AC-READ.
195   >0228                ACCEPT SUBS-LRAD LINE 17 POSITION 45 PROMPT TAB.
                                 $                                    $
*****   1) IDENTIFIER   *E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E *E
*****   2) SCAN RESUME  *W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W
196   >022A                IF SUBS-LREAD NOT NUMERIC OR SUBS-LREAD = ZERO
197                                                    GO TO AC-READ.
198              *
199              *  كتابه السجل على القرص بعد ان تاكدنا من انه صحيح
200              *
201   >0236                WRITE SUBS-REC
202                           INVALID KEY DISPLAY "السجل لم يحفظ الرجاء اعد العمل"
203                              LINE 23 POSITION 10
204                                    ACCEPT DUMMY GO TO EXT-INS.
205   >024C                DISPLAY "الســــــــــجل اضيف"
206                        LINE 23 POSITION 10 ACCEPT DUMMY.
207   >025A       EXT-INS.
208   >025A                EXIT.
209              *
210              *  الفقره التاليه هي للاستفسار عن سجل دون امكانيه تعديله
211              *
212   >025C       INQR-RTN.
213   >025C                    PERFORM ACCEPT-NO.
214   >025E                    IF SUBS-NO = LOW-VALUES GO TO EXT-UPD.
215   >0266                    IF FLAG     = 1
216                                DISPLAY       "الســـجل غير موجود"
```

```
                                            LINE 20 POSITION 10
217                                         ACCEPT  DUMMY GO TO EXT-UPD.
218
219  >027A              PERFORM  DISPLAY-SCREEN.
220  >027C              PERFORM  DISPLAY-FIELDS.
221  >027E              DISPLAY                    اضغط مفتاح الانتهاء للعودة الى الاستفسار جديد
                        "
222                     LINE     23 POSITION 5 ACCEPT DUMMY.
223                     GO TO INQR-RTN.
224  >028A         EXT-INQ.
225  >028C              EXIT.
226  >028C
227            *
228                     الفقرة التالية هى لحذف سجل من الملف بتعليمة
229            *
230            * DELETE
231            *
232  >028E      DELT-RTN.
233  >028E           PERFORM ACCEPT-NO.
234  >0290           IF  FLAG = 1 DISPLAY            "ليس هناك سجل لحذفه فهل تأكد من الرقم"
235                                  LINE 20 POSITION 10
236                                  ACCEPT DUMMY GO TO EXT-DEL.
237                  DISPLAY SUBS-NAME LINE 20 POSITION 10 ERASE
238  >02A4                   "هل انت متأكد من الحذف ، اجب بـ ن او لا "
239                                     LINE 22 POSITION 5.
240                                     LINE 21 POSITON 10 PROMPT TAB.
241  >02B6           ACCEPT ANSR                $                  $
*****  1) IDENTIFIER  *E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E
*****  2) SCAN RESUME *W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W
242  >02B8           IF     ANSR  = "ن" GO TO EXT-DEL.
243                      اذا كانت الاجابة ن اى نعم يتم الحذف كما فى التعليمة التالية
244            *
245            *
246  >02C0           DELETE SUBSFL RECORD.
247  >02CA           DISPLAY "تم حذف السجل"  LINE 23 POSITION 30.
248  >02D2           ACCEPT DUMMY.
249  >02D8      EXT-DEL.
250  >02D8           EXIT.
251            *
```

```
252              * فقرة قبول رقم السجل المراد معالجته ثم قراءة الملف
253              * لمعرفة هل هو موجود أم لا ؟
254  >02DA        ACCEPT-NO.
255  >02DA            MOVE ZERO TO FLAG.
256  >02DE            DISPLAY
257                   "ادخل رقم المشترك المراد العمل عليه او اضغط        للإنتهاء"
258                   LINE 10 POSITION 12 ERASE
259                   "RETURN"
260                   LINE 10 POSITION 19 SIZE 6.
261  >02F2            ACCEPT SUBS-NO LINE 11 POSITION 12 PROMPT TAB.
262  >02FC            IF SUBS-NO = "000000" MOVE LOW-VALUE TO SUBS-NO
263                                                      GO TO EXT-AC.
264  >0308            READ SUBSFL
265                       INVALID KEY MOVE 1 TO FLAG GO TO EXT-AC.
266  >0316            MOVE ZERO TO FLAG.
267  >031C        EXT-AC.
268  >031C            EXIT.
269              * الفقرة التالية هي لعرض الشاشة ثم عرض السجل عليها
270              *
271  >031E        DISPLAY-SCREEN.
272  >031E            DISPLAY
273                    "             شـــــــــــــركة الكهرباء"
274                    LINE 1 POSITION 25 ERASE
275                    "برنامج بناء وتعديل ملف المستهلكين الأساسي"
276                    LINE 3 POSITION 35
277                    "-1 رقم المشترك  "  LINE 3  POSITION 25
278                    "-2 إسم المشترك  "  LINE 5  POSITION 25
279                    "-3عنوان المشترك"  LINE 6  POSITION 25
280                    "-4تاريخ الاشتراك"  LINE 7  POSITION 25
281                    "-5    السنة      "  LINE 8  POSITION 25
282                    "-6    الشهر      "  LINE 9  POSITION 25
283                    "-7    اليوم      "  LINE 10 POSITION 25
284                    "-8  رمز الاشتراك"  LINE 11 POSITION 25
285                    "-9  رمز المنطقة "  LINE 12 POSITION 25
286                    "-10 رقم العداد  "  LINE 13 POSITION 25
287                    "-11  آخر قراءة  "  LINE 14 POSITION 25
288
```

```
 289                 EXT-AC.
                     $    $
*****   1) SYNTAX    *E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E  .*E
*****   1) SCAN RESUME  *W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W
*****   2) DOUBLE DEFINITION  *E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E
 290  >038D                EXIT.
 291
 292                *
 293                 DISPLAY-FIELDS.
                     $
*****   1) SCAN RESUME     *W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W  ]*W
 294  >038D                DISPLAY     SUBS-NO     LINE 5     POSITION 45
 295                                   SUBS-NAME   LINE 5     POSITION 45
 296                                   SUBS-ADRES  LINE 5     POSITION 45
 297                                   SUBS-YEAR   LINE 5     POSITION 45
 298                                   SUBS-MONTH  LINE 5     POSITION 45
 299                                   SUBS-DAY    LINE 5     POSITION 45
 300                                   SUBS-CODE   LINE 5     POSITION 45
 301                                   SUBS-AREA   LINE 5     POSITION 45
 302                                   SUBS-CNTR   LINE 5     POSITION 45
 303  >03DF                            SUBS-LREAD  LINE 5     POSITION 45
 304                 EXT-FLD.
                     $
*****   1) SYNTAX    *E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E
*****   1) SCAN RESUME  *W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W
 305  >03DF                 EXIT.
 306           ZZZZZZ END PROGRAM.                                *** END OF FILE   *
               0                FILE                    SUBSFL
  >0026       80   GRP     0    GROUP                    SUBS-REC
  >0026        5   ANS     0    ALPHANUMERIC              SUBS-NO
  >002B       20   ANS     0    ALPHANUMERIC              SUBS-NAME
  >003F       24   ANS     0    ALPHANUMERIC              SUBS-ADRES
  >0057        8   GRP     0    GROUP                     SUBS-DATE
  >0057        2   NSU     0    NUMERIC UNSIGNED           SUBS-DAY
                           0    CONDITION-NAME              DAYY
  >0059        2   NSU     0    NUMERIC UNSIGNED           SUBS-MONTH
                           0    CONDITION-NAME              MONTHH
  >005B        4   NSU     0    NUMERIC UNSIGNED           SUBS-YEAR
                           0    CONDITION-NAME              YEARR
  >005F        2   NSU     0    NUMERIC UNSIGNED          SUBS-CODE
                           0    CONDITION-NAME              CODEE
```

```
 >0061        2  NSU    0   NUMERIC UNSIGNED          SUBS-AREA
                            0   CONDITION-NAME             AREAA
 >0063        7  NSU    0   NUMERIC UNSIGNED          SUBS-CNTR
 >006A        7  NSU    0   NUMERIC UNSIGNED          SUBS-LREAD
 >007A        2  NBS    0   BINARY SIGNED           DEC-CODE
 >007A        2  ANS    0   ALPHANUMERIC            BIN-VAL
 >007C        2  NSU    0   NUMERIC UNSIGNED        CHAR
 >007E        1  NSU    0   NUMERIC UNSIGNED        FLAG
 >0080        2  NSU    0   NUMERIC UNSIGNED        FLD-NO
 >0082        2  ANS    0   ALPHANUMERIC            DUMMY
 >0084        1  ANS    0   ALPHANUMERIC            ANSR
 >0086        1  NSU    0   NUMERIC UNSIGNED        OPTIONS
ILLEGAL PERFORM *E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E UPDAT-RTN
                                            THRU EXT-UPD
ILLEGAL PERFORM *E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E AC-ADRES
ILLEGAL PERFORM *E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E AC-LREAD
ILLEGAL PERFORM *E*E*E*E*E*E*E*E*E*E*E*E*E*E*E*E DISPLAY-SCREEN
                                            THRU EXT-SC
READ ONLY BYTE SIZE =       >0940
READ/WRITE BYTE SIZE =      >0130
OVERLAY SEGMENT BYTE SIZE = >0000
TOTAL BYTE SIZE =           >0A70
   12 ERRORS
    9 WARNINGS.
```



استخدام الحاسب في كتابة وترجمة وتنفيذ برنامج كوبول

الفصل الحادي عشر

استخدام منقح النصوص
TEXT EDITOR

١١ ـ ١ لمحة عامة :

يعتبر منقح النصوص TEXT EDITOR من البرامج الأساسية الموجودة في نظام تشغيل أي حاسب آلي يستخدم لغة كوبول.

ويستخدم منقح النصوص كوسيلة لادخال البرامج ـ الملفات(*) ـ إلى الحاسب، أو تعديل برامج ـ ملفات ـ قديمة.

إن تقدم التقنية الخاصة بأجهزة الحاسب HARDWARE مع تقدم أنظمة التشغيل والبرامج الأساسية SOFTWARE جعلت استخدام منقح النصوص عبر النهائيات ـ الشاشات ـ TERMINALS اسهل تقريباً، وذلك بتقديمها مجموعة أوامر وقواعد ساعدت كثيراً على عملية تحديث البرامج ـ الملفات ـ وسنقدم في هذا الفصل عرضاً لاستخدام برنامج منقح النصوص وفق نظامي تشغيل مختلفين. الأول على جهاز IBM 3033S ذي نظام تشغيل يسمى MVS والثاني على جهاز TEXAS INSTRUMENTS ذي نظام التشغيل DX10. وسنقدم بالبداية لمحة سريعة عن كيفية تشغيل النهائيات للوصول إلى برنامج منقح النصوص مع ذكر العمليات الأولية الواجب اتخاذها عند استخدام جهاز IBM للمرّة الأولى.

(*) توجد قيود على طول السجل في الملف تختلف باختلاف الجهاز أو النظام المستخدم، فمثلا IBM MVS يصل طول ٢٥٥ بايت أما TI990 فيصل إلى ٨٠ بايت.

١١ ـــ ٢ استخدام نظام الحاسب IBM 3033

١١ ـ ٢ ـ ١ العمليات الواجب اتباعها عند العمل على جهاز IBM 3033 للمرة الأولى :

هناك عدة اجراءات يجب اتخاذها قبل تنفيذ أي برنامج، هذه العمليات أو الاجراءات تتضمن إعطاء قيم معينة يستخدمها الحاسب كل مرّة ونحددها نحن فقط مرّة واحدة، وتتناول هذه العمليات تحديد نوع الشاشة المستخدمة والطابعة المستخدمة، والاسم الخاص بالمستخدم USER NAME وغيرها من العمليات.

وقبل البدء بتحديد هذه المعطيات نشير إلى أنه هذه المعطيات تختلف من موقع SITE إلى آخر، وسنتحدث عن القيم المعطاة في مركز الحاسب الآلي لمعهد الادارة العامة.

(أ) عند فتح الشاشة وطلب برنامج SPF تأتي الشاشة رقم ١١ ـــ ١ نطلب الخيار 0 صفر فتظهر لنا الشاشة رقم ١١ ـــ ٢ وفيها ثلاثة خيارات OPTIONS 3، نطلب الخيار رقم ١ وهو خاص بتوصيف الشاشة نفسها، فتظهر لنا الشاشة رقم ١١ ـــ ٣ ونطبق القيم المختلفة فيها مع محتوى الشاشة رقم ١١ ـــ ٣ المرفقة. ثم نضغط مفتاح PF15 للعودة للشاشة رقم ١١ ـــ ١
نطلب الخيار 2 في الشاشة رقم ١١ ـــ ٢ وهو خاص بتحديد قيم وحدات الطباعة المستخدمة، فتظهر لنا الشاشة رقم ١١ ـــ ٤ نطابق محتوى الشاشة رقم ١١ ـــ ٤ مع ما يظهر على الشاشة ونعد له مع ملاحظة ما يلي :ـــ

١ ـــ يكتب رقم التعريف USER-ID مكان الأحرف XXXX

٢ ـــ يكتب اسم التعريف مكان الأحرف YYY

٣ ـــ يكتب الاسم مكان الأحرف 'NNNNN'

٤ ـــ ونضع الحرف (*) H عند التعبير = CLASS
٥ ـــ يوضع USER-ID كاملاً عند التعبير = NOTIFY

نكرر هذه الملاحظات بالنسبة للشاشة رقم ١١ ـــ ٥.
بعد ادخال هذه المعايير نضغط مفتاح PF15 للانتهاء والعودة للشاشة رقم ١١ ـــ ٢.
يمكن طلب الخيار رقم ١١ ـــ ٣ في الشاشة رقم ١١ ـــ ٢ للاطلاع على مفاتيح التحكم. اضغط مفتاح PF15 ـــ أكثر من مرّة إذا لزم الأمر ـــ للعودة للشاشة الأساسية رقم ١١ ـــ ١.

(ب) تحديد شكل ومكان طباعة المخرجات OUTPUT LIST

قد نحتاج لطباعة برنامج SOURCE PROGRAM أو ملف معين، لذلك نطلب الخيار رقم 3.6 اعتباراً من الشاشة الأساسية رقم ١١ــ١ فتظهر لنا الشاشة رقم ١١ــ٦. نكرر الملاحظات السابقة في تعبئة قيم لغة التحكم JCL الواردة في الفقرة أ نضغط مفتاح PF15 للعودة للشاشة الأساسية رقم ١١ــ١.

(*) يختلف باختلاف مركز الحاسب.

SPF-MVS PRIMARY OPTION MENU

SELECTION OPTION ===>

0	SPF PARMS	— SPECIFY TERMINAL AND SPF PARAMENTERS
1	BROWSE	— DISPLAY SOURCE DATA OR OUTPUT LISTINGS
2	EDIT	— CREATE OR CHANGE SOURCE DATA
3	UTILITIES	— PERFORM SPF UTILITY FUNCTIONS
5	BACKGROUND	— COMPILE, ASSEMBLE, OR LINK EDIT
6	COMMAND	— ENTER TSO COMMAND OR CLIST
T	TUTORIAL	— DISPLAY INFORMATION ABOUT SPF
X	EXIT	— TERMINATE SPF USING LIST / LOG DEFAULTS

PRESS END KEY TO TERMINATE SPF

INSTITUTE
OF
PUBLIC ADMINISTRATION

شكل رقم (١١ ـ ١)

SPF PARAMETER OPTION

SELECTION OPTIONS ===>

1	TERMINAL	— SPECIFY TERMINAL CHARACTERISTICS
2	LOG/LIST	— SPECIFY SPF LOG AND LIST DEFAULTS
3	PF KEYS	— SPECIFY PF KEYS FOR 3278BL TERMINAL WITH 24 PF KEYS

شكل رقم (١١-٢)

TERMINAL CHARACTERISTICS

ENTER/VERIFY PARAMETERS BELOW :

TERMINAL TYPE	===>	3278BL	(3277 — 3277 / 3275 DISPLAY) (3278 — 3278 / 3276 DISPLAY) (3278BL — 3278 / 3276 DISPLAY BILING-UAL ARABIC)
NUMBER OF PF KEYS	===>	24	(12 — TWELVE PROGRAM FUNCTION KEYS) (24 — TWENTY FOUR PROGRAM FUNC-TION KEYS)
INPUT FIELD PAD	===>	N	(N — NULLS) (B — BLANKS)
SCREEN FORMAT (3278 MODEL 5 ONLY)	===>	DATA	(DATA — FORMAT BASED ON DATA WIDTH) (STD — ALWAYS FORMAT 24 LINES BY 80 CHARS) (MAX — ALWAYS FORMAT 27 LINES BY 132 CHARS)

شكل رقم (١١ـ٣)

LOG AND LIST DEFAULTS

ENTER / VERIFY PARAMETERS BELOW :

LOG DATASET DEFAULT OPTIONS			LIST DATASET DEFAULT OPTIONS		
PROCESS OPTION	===>	D	PROCESS OPTION	===>	J
SYSOUT CLASS	===>	R	SYSOUT CLASS	===>	M
LOCAL PRINTER ID	===>		LOCAL PRINTER ID	===>	LP54E
LINES PER PAGE	===>	60	LINES PER PAGE	===>	60
PRIMARY PAGES	===>	0	PRIMARY PAGES	===>	100 **
SECONDARY PAGES	===>	0	SECONDARY PAGES	===>	200 **

(**) EFFECTIVE NEXT SESSION

VALID PROCESS OPTIONS :

J — SUBMIT JOB TO PRINT (AND DELETE)

L — ROUTE TO LOCAL PRINTER (AND DELETE)

D — DELETE DATASET (DO NOT PRINT)

JOB STATEMENT INFORMATION :(IF OPTION "J" SELECTED)

```
===> //  STDxxxx JOB (STD, xxxx), 'NNNNN', CLASS = H, MSGCLASS = Y,
===> //          NOTIFY = STD xxxx
===> // *
===> // *
```

شكل رقم (١١-٤)

SPECIFY DISPOSITION OF LIST DATASET ____________________

ENTER / VERIFY PARAMETERS BELOW :

NO LOG DATASET THIS SESSION

LIST OPTIONS FOR THIS SESSION

PROCESS OPTION === J

SYSOUT CLASS === M

LOCAL PRINTER ID === LP54E

VALID PROCESS OPTIONS :

J — SUBMIT JOB TO PRINT (AND DELETE)

K — KEEP DATASET (DO NOT PRINT)

L — ROUTE TO LOCAL PRINTER (AND DE LETE)

D — DELETE DATASET (DO NOT PRINT)

PRESS ENTER TO COMPLETE SPF TERMINATION

PRESS END KEY TO RETURN TO PRIMARY OPTION MENU

JOB STATEMENT INFORMATION : (IF OPTION "J" SELECTED)

```
===>   //STDxxxxJOB(STD,xxxx),'NNNN',CLASS = H,MSGCLASS=Y,
===>   //     NOTIFY=STDxxxx
===>   //*
===>   //*
```

شكل رقم (١١ــ٥)

```
———————————— HARDCOPY UTILITY ————————————

SELECTION OPTION  ===>   L

   J — GENERATE JCL TO PRINT OR PUNCH DATASET
   L — ROUTE DATASET TO LOCAL PRINTER


DATASET NAME ===>
     DISPOSITION          ===>    KEEP        (KEEP OR DELETE)
     VOLUME SERIAL        ===>                (IF NOT CATALOGED)
     DATASET PASSWORD     ===>                (IF PASSWORD PROTECTED)

SYSOUT CLASS        ===>   M                  (IF OPTION "J" SELECTED)
LOCAL PRINTER ID    ===>   LP54E              (IF OPTION "L" SELECTED)

JOB STATEMENT INFORMATION : (IF OPTION "J" SELECTED, VERIFY BEFORE
                              PROCEEDING)
  ===>    //STDxxxxJOB(STD,xxxx), 'NANN' CLASS =H,MSGCLAS=Y,
  ===>    //     MSGLEVEL  =  (2,1)
  ===>    //٥
  ===>    //٥
```

شكل رقم (١١ — ٦)

١١ ـ ٢ ـ ٢ تشغيل النهائيات من نوع IBM 3278

يتم تشغيل هذا النوع من الشاشات أو النهائيات بالشكل التالي :

ــ نضع المفتاح الأحمر الموجود على يسار الشاشة على الوضع I

ــ ننتظر قليلاً حتى تظهر أمامنا الشاشة والضوء الأحمر

ــ نضغط مفتاح ALT مع مفتاح ATTN بنفس الوقت .

ــ نكتب بعدها عبارة TSO أي TIME SHARING OPTION

ــ ننتظر قليلاً وتظهر لنا الرسالة التالية :ــ

ــ * ENTER USERID ــ

أي أدخل عبارة التعريف الخاصة بنا للحاسب ، نكتب عندها عبارة التعريف الخاصة بنا ولتكن STD4221 مثلاً وننتظر قليلاً .

ــ تظهر لدينا عدّة رسائل لا تعيننا بشيء غالباً .

ــ ننتظر حتى تظهر لنا عبارة .

READY

عندها يكون الحاسب مستعداً لتلقي الأوامر منا .

ــ نكتب بعدها عبارة SPF ونضغط مفتاح ENTER

وتعني STRUCTURE PROGRAM FACILITIES

ــ ننتظر قليلاً وتظهر لنا ثلاث نجوم *** نضغط مفتاح ENTER .

ــ ننتظر قليلاً لتظهر لنا الشاشة الرئيسية في هذا البرنامج أنظر الشكل المرفق رقم ١١ــ١ .

انطلاقاً من هذه الشاشة باستطاعتنا كتابة برنامج وطباعته والتعديل عليه واستغلال كثير من التسهيلات التي يقدمها الحاسب .

(*) من الممكن أن يكون هناك كلمة سر PASSWORD فتدخل عند ظهور الرسالة المناسبة .

فعند كتابة أي رقم في الخانة المخصصة لذلك وهي SELECT OPTION تأتينا شاشة ثانية ومنها شاشة فرعية ثالثة... الخ.

اغلاق النهائيات ــ الشاشات من نوع IBM3278

ــ أينما كنا نضغط مفتاح انتهاء PF15 للوصول إلى الشاشة رقم ١١ــ١.

ــ بعد الوصول للشاشة رقم ١١ــ١ نكتب حرف × في مكان الاختيار SELECT OPTISON ونضغط ENTER

ــ ننتظر قليلاً لتظهر لنا عبارة READY

ــ نكتب تعليمة LOGOFF ونضغط ENTER

ــ ننتظر قليلاً بعدها نطفيء مفتاح التشغيل الأحمر، أي نرجعه للوضع 0

ــ لا تنسى أوراقك الخاصة قرب الشاشات لأنها قد تعتبر أوراقاً عديمة القيمة وتتلف.

ــ حافظ على نظافة المكان قبل مغادرتك صالة الحاسب.

مجموعة أزرار التحكم FUNCTION KEYS

يوجد على يمين لوحة الأزرار KEY BOARD مجموعة من المفاتيح معنونة كالتالي: PF13 , PF14 ,..... PF24 وكذلك أعلى مفاتيح الأرقام مجموعة من أزرار التحكم PF1 , PF2 , PF3 ... تستخدم مع مفتاح ALT

ما يهمنا منها هي مجموعة أزرار التحكم من 13 إلى 24 لأن مفاتيح التحكم من (١) وحتى (١٢) مطابقة في عملها للمفاتيح من 13 إلى 24 انظر للجدول التالي لتلاحظ عمل هذه المفاتيح: الشكل رقم ١١ــ٧.

PROGRAM FUNCTION KEY DEFINITION

ENTER / VERIFY PARAMETERS BELOW FOR 3278T TERMINAL WITH 24 PF KEYS :

PF1 ==>	HELP	PF13 ==> HELP	للمساعدة
PF2 ==>	SPLIT	PF14 ==> SPLIT	لفصل الشاشة إلى اثنين
PF3 ==>	END	PF15 ==> END	لانهاء عمل
PF4 ==>	RETURN	PF16 ==> RETURN	للعودة للشاشة الأساسية
PF5 ==>	FIND	PF17 ==> FIND	لايجاد حرف أو مجموعة حروف
PF6 ==>	CHANGE	PF18 ==> CHANGE	لتغيير حرف أو مجموعة حروف
PF7 ==>	UP	PF19 ==> UP	لارجاع ما عرض على الشاشة للخلف
PF8 ==>	DOWN	PF20 ==> DOWN	لتقديم الشاشة للأمام
PF9 ==>	SWAP	PF21 ==> SWAP	للانتقال من شاشة للثانية
PF10 ==>	LEFT	PF22 ==> LEFT	لتحريك محتوى الشاشة لليسار
PF11 ==>	RIGHT	PF23 ==> RIGHT	لتحريك محتوى الشاشة لليمين
PF12 ==>	CURSOR	PF24 ==> CURSOR	

VALID OPTIONS : (BLANK RESETS TO DEFAULT OPTION)

HELP SPLIT END RETURN FIND CHANGE

UP DOWN SWAP LEFT RIGHT CURSOR

NOP PRINT PRINT_HI

OR EQUATE TO EDIT / BROWSE COMMAND :

>CMD ... (PRIMARY COMMAND) EXAMPLE : PF10 ==>> TABS ON

: CMD ... (EDIT LINE COMMAND) EXAMPLE : PF11 == > : I

شكل رقم (١١ـ٧)

١١ - ٣ استخدام منقح النصوص TEXT EDITOR من خلال شاشات IBM 3278

١١ - ٣ - ١ كتابة برنامج جديد :

انطلاقاً من الشاشة الرئيسية رقم ١١ــ٨ نكتب الخيار 2 وهو EDIT، هذا الخيار يتيح لنا كتابة برنامج جديد بلغة الكوبول أو تعديل برنامج سبق كتابته فعند إدخال ٢ والضغط على مفتاح ENTER تظهر لدينا الشاشة رقم ١١ــ٩.

ونتأكد من أنه حقل PROJECT يحتوي على STDxxxx (*)

وأن حقل LIBRARY يحتوى على TEST

وأن حقل TYPE يحتوى على COBOL

بعد التأكد نكتب اسم البرنامج الجديد بحقل MEMBER ولنفترض أنه PROL نضغط مفتاح ENTER فيفتح ملف جديد باسم PROL ضمن المكان المخصص للكتابة DATASET انظر الشاشة رقم ١١ــ١٠.

نضغط على مفتاح NEW LINE لنصل للسطر الأول ونكتب عبارة COLS لتحديد المواقع التي نرغب بالعمل عليها. فنحن عادة في الكوبول نبتدىء بالعمود الثامن كما مرّ معنا في الفصل الأوّل. نضغط ENTER لنحصل على الشاشة رقم ١١ــ١١ نبدأ بالعمود رقم ٨ بكتابة البرنامج المعطى لنا كما هو واضح في الشاشات رقم ١١ــ٥، ١١ــ٦، ١١ــ٧، ١١ــ٨.

يلاحظ من الأشكال أن سطر نهاية الملف BOTTOM OF DATA يظهر عقب السطر الأول وللتغلب على ذلك نرجع لأعلى الشاشة بضغط مفتاح PF24 بعدها نضغط

(*) هذه الأسماء تتبدل من موقع لآخر، لذلك راجع المشرف بشأن إعطاء الاسم المناسب.

مفتاح NEW LINE لنصل للسطر الذي ترغب بادخال أسطر جديدة بعده نكتب الأمر I أي INSERT و يعني ادخل أسطرا بعد هذا السطر فاذا كتبنا I20 ــ أي أدخل أو افتح ٢٠ سطراً جديداً ــ نضغط ENTER لنحصل على صفحة كاملة لادخال البرنامج انظر الشاشة رقم ١١ــ١٤ والسهم الدال على مكان كتابة التعليمة I20 وكذلك الشكل رقم ١١ــ١٥.

عند الانتهاء من كتابة البرنامج نضغط مفتاح PF15 لحفظ برنامج تحت اسم PROL مثلاً نضغط PF15 مرّة ثانية لنعود إلى الشاشة الرئيسية رقم ١١ــ٨.

SPF-MVS PRIMARY OPTION MENU

SELECTION OPTION ===>

0	SPF PARMS	_ SPECIFY TERMINAL AND SPF PARAMENTERS
1	BROWSE	_ DISPLAY SOURCE DATA OR OUTPUT LISTINGS
2	EDIT	_ CREATE OR CHANGE SOURCE DATA
3	UTILITIES	_ PERFORM SPF UTILITY FUNCTIONS
4	FOREGROUND	_ COMPILE, ASSEMBLE, LINK EDIT, OR DEBUG
5	BACKGROUND	_ COMPILE, ASSEMBLE, OR LINK EDIT
6	COMMAND	_ ENTER TSO COMMAND OR CLIST
7	SUPPORT	_ TEST DIALOG OR CONVERT MENU / MESSAGE FORMATS
T	TUTORIAL	_ DISPLAY INFORMATION ABOUT SPF
X	EXIT	_ TERMINATE SPF USING LIST / LOG DEFAULTS

PRESS END KEY TO TERMINATE SPF

INSTITUTE
OF
PUBLIC ADMINISTRATION

شكل رقم (١١ ــ ٨)

```
———————————— EDIT — ENTRY PANEL ————————————

ENTER / VERIFY PARAMETERS BELOW :


SPF LIBRARY :

   PROJECT  ===>   STD xxxx

   LIBRARY  ===>   TEST     ===>     ===>     ===>

   TYPE     ===>   COBOL

   MEMBER   ===>              (BLANK FOR MEMBER SELECTION LIST)


OTHER PARTITIONED OR SEQUENTIAL DATASET :

   DATASET NAME      ===>

   VOLUME SERIAL     ===>          (IF NOT CATALOGED)


   DATSET PASSWORD   ===>          (IF PASSWORD PROTECTED)


   PROFILE NAME      ===>          (BLANK DEFAULTS TO DATASET TYPE)
```

شكل رقم (٩ــ١١)

```
EDIT _ PAY. TEST. COBOL (PRG1)                              COLUMNS 001 072
COMMAND INTPUT ===>                                   SCROLL ===>    HALF
******    ********************* TOP OF DATA **
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
******    **************** BOTTOM OF DATA *********
```

شكل رقم (١١ــ١٠)

```
EDIT — PAY. TEST. COBOL (PRG1)                              COLt
COMMAND INTPUT ===                                          SCROLL ==
******      ****************** TOP OF DATA ***********
= COLS      —+—1—+—2—+—3—+—4—+—5—+—6—+—
......
******      **************** BOTTOM OF DATA *********
```

شكل رقم (١١ـ١١)

```
EDIT — PAY. TEST. COBOL (PRG1)                                   COLI
COMMAND INTPUT ===                                               SCROLL =
******   ****************** TOP OF DATA ***********
= COLS   —+—1—+—2—+—3—+—4—+—5—+—6—+—
......              IDENTIFICATION DIVISION.
......
......
......
......
......
......
......
......
......
......
......
......
......
......
......
......
......
......
......
```

شكل رقم (١١ــ١٢)

```
EDIT — PAY. TEST. COBOL (PRG1)                                   COLU
COMMAND INTPUT ===                                          SCROLL =
******      ****************** TOP OF DATA ***********
= COLS      —+—1—+—2—+—3—+—4—+—5—+—6—+—
000100              IDENTIFICATION DIVISION.
******      **************** BOTTOM OF DATA *********
```

شكل رقم (١١ــ١٣)

```
EDIT — PAY. TEST. COBOL (PRG1)
COMMAND INTPUT ===                                          SCROLL =
******      ****************** TOP OF DATA ***********
= COLS      —+—1—+—2—+—3—+—4—+—5—+—6—+—
I20100              IDENTIFICATION DIVISION.
******      **************** BOTTOM OF DATA *********
```

شكل رقم (١١ــ١٤)

```
EDIT — PAY. TEST. COBOL (PRG1)                          COLU
COMMAND INTPUT ===                                      SCROLL ==
******   ooooooooooooooooooo TOP OF DATA ooooooooooo
= COLS   —+—1—+—2—+—3—+—4—+—5—+—6—+—
000100          IDENTIFICATION DIVISION.
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
```

يظهر الأسطر الجديدة بعد تنفيذ أمر I20

شكل رقم (١١ــ١٥)

١١ ــ ٣ ــ ٢ تنقيح برنامج ــ ملف ــ موجود سابقاً *

إن إدخال البرنامج إلى الحاسب لا يعتبر نهاية العمل بالنسبة لهذا البرنامج فهو قد يحتوي على أخطاء قاعدية SYNTAX ERROR ناتجة إما عن خطأ في إدخال عبارات البرنامج؛ أو عن صياغة هذه العبارات أصلاً.

مهما يكن من أمر فان متطلبات صيانة البرامج تفرض تعديل هذا البرنامج ولو كان خالياً من الأخطاء القاعدية.

ومن أجل تعديل برنامج معين في نظام IBM3033 باستخدام نهائيات من نوع 3278 نتبع ما يلي:

اعتباراً من الشاشة الأساسية رقم ١١ ــ ١ نطلب الخيار رقم ٢ وهو برنامج EDIT فتأتي الشاشة رقم ١١ ــ ١٦ نكتب اسم البرنامج الذي نريد التعديل عليه، نضغط مفتاح ENTER فتأتي صفحة جديدة تحتوى على البرنامج المراد العمل عليه.

إن عمليات التعديل قد تتناول سطرا أو أسطرا معينة بالبرنامج أو قد تتناول عبارة محددة متكررة في البرنامج، أو تعالج البرنامج ككل لذلك سنقسم شرح طريقة العمل على منقح النصوص TEXT EDITOR إلى الأقسام التالية** :

أ ــ الأوامر التي تعالج الأسطر LINE COMMANDS

ب ــ أوامر أساسية عامة GENERAL COMMANDS

جـ ــ أوامر البحث عن حرف أو سلسلة أحرف، أوامر تبديل حرف أو سلسلة أحرف FINDING AND CHANGING COMMANDS

د ــ أوامر متقدمة.

(*) يحتوى برنامج EDIT العديد من الأوامر الاضافية المتقدمة لذا من أجل زيادة الاطلاع يرجع إلى IBM MANUALS.

(**) Time Sharing Option 3270 Display Support and Structured Programming Facility Version 202 : Program Reference Manual.

```
________________  EDIT _ ENTRY PANEL  ________________

ENTER / VERIFY PARAMETERS BELOW :

SPF LIBRARY :
   PROJECT  ===>   STD4253
   LIBRARY  ===>   TEST    ===>        ===>        ===>
   TYPE     ===    COBOL
   MEMBER   ===>          (BLANK FOR MEMBER SELECTION LIST)

OTHER PARTITIONED OR SEQUENTIAL DATASET :
   DATASET NAME   ===>
   VOLUME SERIAL  ===>      (IF NOT CATALOGED)

DATASET PASSWORD  ===>      (IF PASSWORD PROTECTED)

PROFILE NAME      ===>      (BLANK DEFAULTS TO DATASET TYPE)
```

شكل رقم (١١ ــ ١٦)

والجدول التالي يلخص هذه الأ وامر:

	أوامر سطرية LINE COMMANDS	أوامر أساسية Primary Commands
Basic Line Commands أوامر الأسطر الأساسية	COLS, M-MOVE I-INSERT, C-COPY D-DELETE, A-AFTER R-REPEAT, B-BEFORE	
General Command أوامر عامة		**Locate Print** **Number Reset** **RENUM Submit** **UNNUM Save** **Autonum Cancel**
Find, Change أمر البحث والتبديل		FIND CHANGE
أوامر متقدمة Advanced Features	(— COLS LEFT) — COLS RIGHT	**Copy, Profile** **Move HEX** **Create Replace**

قبل البدء في استعراض هذه الأوامر نذكر أنه يمكن تحريك إشارة التوضح — الدليل CURSOR إلى أي مكان على الشاشة باستخدام أسهم التحريك الأربعة الموجودة على لوحة المفاتيح KEYBOARD

(أ) الأ وامر الأساسية التي تعالج أسطر BASIC LINE COMMANDS

هذه الأ وامر هي :

تحديد المواقع ــ الأعمدة COLS (COLUMNS)

إدخال سطر أو أسطر جديدة I (INSERT)

حذف سطر أو أكثر D (DELETE)

تكرار سطر R (REPEAT)

نقل سطر من مكان لآخر M (MOVE)

نسخ سطر C (COPY)

بعــد A (AFTER)

قبــل B (BEFORE)

جميع هذه الأ وامر تكتب اعتباراً من الموقع POSITION الأ ول من السطر ــ أو الأسطر ــ المراد العمل عليها وسنسميها منطقة الأ وامر السطرية. وتكتب الأ وامر فوق الأ رقام الموجودة أصلاً في السطر.

١ ــ أمر COLS

هذا الأمر يحدد الأعمدة أو المواقع (POSITIONS) للسطر، و يفيد هذا الأمر لمعرفة الموقع الذي يجب أن تبدأ فيه كتابة السطر، ففي برنامج كو بول يجب أن تبدأ الفقرات وعناو ين الأقسام DIVISIONS و SECTIONS في الهامش أ MARGIN A أي من الموقع الثامن إلى الحادي عشر. بينما التعليمات STATEMENTS في الهامش ب MARGIN B أي من العمود ١٢ إلى ٧٢. وتوضح النجمة (*) في العمود السابع للدلالة على ان هذا السطر مخصص للملاحظات.

ويمكن كتابة السطر الأ ول باستخدام أمر COLS بعدها تكتب باقي الأسطر بالقياس. انظر الشكل رقم (١١ــ١٧).

```
EDIT — STD4253. TEST. COBOL (PROGA) — 01.20                COLUMNS 001 072
COMMAND INTPUT ===>                                  SCROLL ===>    HALF
******  ****************** TOP OF DATA ********************
000100   INDENTIFICATION DIVISION
= COLS   —+—1—+—2—+—3—+—4—+—5—+—6—+—7—
000200          PROGRAM-ID. PROGA.
000300    *   THIS PROGRAM PRINTS IPA—
000400    *   NAME AND ADDRESS
000500    *
000600          ENVIRONMENT DIVISION.
000700          CONFIGURATION SECTION.
000800          SOURCE-COMPUTER. IBM-370.
000900          OBJECT-COMPUTER. IBM-370.
001000    *
001100          INPUT-OUTPUT SECTION.
001200          FILE-CONTROL.
001300          SELECT CARD-F ASSIGN TO UT-S-CARD.
001400          SELECT PRINT-F ASSIGN TO UT-PRFL.
001500    *
001600          DATA DIVISION.
001700          FILE SECTION.
001800          FD CARD-F RECORD CONTAINS 80 CHARACTERS
001900          LABEL RECORD IS OMITTED.
002000          01 CARD-REC PIC X (80).
```

شكل رقم (١١ــ١٧)

٢ ــ أمر INSERT إدخال سطر (I)

يستخدم هذا الأمر لادخال سطر أو أكثر إما في نهاية البرنامج ــ الملف ــ أو بين سطرين معينين ضمن البرنامج.

و يكتب الحرف In في السطر الذي يراد إضافة أسطر جديدة بعده ثم يضغط مفتاح ENTER فتحصل إزاحة بقدر سطر ــ أو أكثر حسب قيمة n ــ تتيح إدخال الأسطر المطلوبة. هذه الأسطر الجديدة تعامل كأي سطر قديم موجود أصلاً وتعطى رقما متسلسلا.

بعد إدخال السطر الجديد نضغط ENTER نجد أن فراغاً آخراً ــ ازاحة ثانية ــ قد تمت تسمح بادخال سطر جديد آخر، فاذا لم يكن هناك أي أسطر جديدة يمكن ضغط مفتاح ENTER مرة ثانية لانهاء أمر I المحدث سابقاً.

انظر الشكل رقم (١١ــ١٨) والشكل رقم (١١ــ١٩)

٣ ــ أمر حذف سطر DELETE D

قد نحتاج أحياناً لحذف سطر أو أكثر من النص. فاذا أردنا حذف سطر معين ما علينا إلا وضع حرف D في بداية السطر المراد حذفه وضغط مفتاح ENTER ليحذف هذا السطر.

أما عندما نريد حذف مجموعة أسطر متسلسلة فنكتب عبارة DD في بداية الأسطر المراد حذفها و DD في السطر الأخير من المجموعة. وعند ضغط ENTER تحذف مجموعة الأسطر هذه.

كما يمكن الحذف بطريقة أخرى، وذلك بأن نضع الأمر D وأمامه عدد الأسطر المطلوب حذفها Dn وعند ضغط مفتاح ENTER تحذف هذه الأسطر.

انظر الشكل رقم (١١ــ٢٠)

```
EDIT — STD4253. TEST. COBOL (PROGA) — 01.20 ———————— COLUMNS 001 072
COMMAND INTPUT ===>                                   SCROLL ===>   HALF
003100    *
003200              PERFORM READ-WRITE THRU EXT
003300              UNTIL END-OF-JOB  =  'YES'.
003400              CLOSE CARD-F PRINT-F.
I03500              STOP RUN.
003600    *
003700        READ-WRITE.
003800              READ CARD-F AT END MOVE 'YES' TO
003900              END-OF-JOB GO TO EXT.
004000    *
004100              MOVE CARD-REC TO PRINT-LINE.
004200              WRITE PRINT-LINE AFTER 1 LINE.
004300    *
004400      EXT.
004500             EXIT.
******      ***************** BOTTOM OF DATA  *****************
```

شكل رقم (١١ـ١٨)

```
EDIT — STD4253. TEST. COBOL (PROGA) — 01.20 ———————— COLUMNS 001 072
COMMAND INTPUT ===>                                   SCROLL ===>    HALF
003100    *
003200            PERFORM READ-WRITE THRU EXT
003300            UNTIL END-OF-JOB  = 'YES'.
003400            CLOSE CARD-F PRINT-F.
003500            STOP RUN.
''''''
            _
003600    *
003700       READ-WRITE.
003800            READ CARD-F AT END MOVE 'YES' TO
003900            END-OF-JOB GO TO EXT.
004000    *
004100            MOVE CARD-REC TO PRINT-LINE.
004200            WRITE PRINT-LINE AFTER 1 LINE.
004300    *
004400    EXT.
004500         EXIT.
******    ***************** BOTTOM OF DATA *****************
```

شكل رقم (١١ــ١٩)

```
EDIT — STD4253. TEST. COBOL (PROGA) — 01.20
COMMAND INTPUT ===>  .                          SCROLL ===>   HALF
003100   *
003200            PERFORM READ-WRITE THRU EXT
003300            UNTIL END-OF-JOB  = 'YES'.
003400            CLOSE CARD-F PRINT-F.
003500            STOP RUN.
DD3600   *
003700        READ-WRITE.
003800            READ CARD-F AT END MOVE 'YES' TO
003900            END-OF-JOB GO TO EXT.
DD4000   *
004100            MOVE CARD-REC TO PRINT-LINE.
004200            WRITE PRINT-LINE AFTER 1 LINE.
004300   *
004400   EXT.
004500          EXIT.
******   ***************** BOTTOM OF DATA *****************
```

شكل رقم (١١ ــ ٢٠)

٤ ــ أمر التكرار REPEAT : R

هذا الأمر يفيد بتكرار سطر معين مرّة أو عدة مرات مباشرة بعد السطر المكتوب عليه هذا الأمر.

فاذا كتبنا R في السطر رقم ٢٠ و بعد ضغط ENTER يتكرر السطر رقم ٢٠ مرّة ثانية؛ ولو أردنا تكرار السطر عدد من المرات نكتب Rn حيث ان n عدد مرات التكرار.

ويمكن أن نكرر مجموعة من الأسطر BLOCK OF LINES عدد من المرات وذلك بكتابة الأمر RR في السطر الأول من المجموعة و RR في السطر الأخير من المجموعة و بعد ضغط ENTER تتكرر هذه المجموعة مرّة ثانية مباشرة بعد المجموعة الأساسية ويمكن أن تتكرر n مرّة RRn .

انظر الشكل رقم (١١ ــ ٢١) والشكل رقم (١١ ــ ٢٢).

٥ ــ أمر MOVE : M

هذا الأمر يسمح بنقل سطر أو مجموعة أسطر من مكان معين في البرنامج ــ الملف ــ إلى مكان آخر.

و يستخدم بوضع الأمر M في السطر المراد نقله ثم الانتقال إلى المكان الأخر المراد نقل السطر إليه نضع الأمر A أو B عند السطر المناسب و يعني الأمر A أن يتم نقل السطر المميز بالأمر M إلى مكان آخر من البرنامج إلى ما بعد السطر المميز بالأمر A (AFTER)، أما إذا وضعنا الأمر B في سطر معين فهذا يعني أننا نريد نقل السطر المميز بالأمر M لما قبل BEFORE السطر المميز بالأمر B .

كما ويمكن نقل مجموعة من الأسطر لمكان آخر من البرنامج وذلك بكتابة الأمر MM في السطر الأول من المجموعة أو MM في السطر الأخير من المجموعة مع كتابة B أو A في مكان آخر من البرنامج، وعند ضغط مفتاح ENTER تنتقل مجموعة الأسطر من

مكانها إلى ما بعد A ، أو ما قبل B السطر المميز بالحرف A أو B .

انظر الشكل رقم (١١ــ٢٣)

٦ ــ أمر COPY : C

يستخدم هذا الأمر لنسخ سطر أو مجموعة أسطر من مكان معين في البرنامج ــ الملف ــ إلى مكان آخر.

وعمله شبيه جداً بعمل الأمر MOVE إلا أن الأخير COPY يطبع نسخه من السطر أو الأسطر بمكان آخر من البرنامج بينما MOVE ينقل السطر أو الأسطر من مكان معين في البرنامج إلى مكان آخر و يستخدم الأمران B أو A بالطريقة السابقة نفسها من أجل نسخ السطر أو الأسطر قبل أو بعد سطر معين.

كما ويمكن نسخ مجموعة أسطر بالأمر CC كما هو وارد في الأمر MM مثال :

لو أردنا نسخ الأسطر من رقم (٣٦١٠) إلى (٣٦٤٠) بمكان آخر بعد السطر (٣٩٠٠) في البرنامج.

نكتب الأمر CC في السطر ٣٦١٠ وكذلك الأمر CC في السطر (٣٦٤٠) ونذهب للسطر ٣٩٠٠ ونكتب عليه الأمر A نضغط مفتاح ENTER فتنسخ الأسطر من ٣٦١٠ إلى ٣٦٤٠ بعد السطر رقم ٣٩٠٠ وتأخذ تسلسلاً جديداً وتبقى الأسطر من ٣٦١٠ ــ ٣٦٤٠ كما هي عليه.

انظر المثال شكل رقم (١١ ــ ٢٤)

كما ويمكن نسخ سطر معين عددا من المرات وذلك بكتابة Cn بحيث تدل n على عدد المرات المطلوبة لنسخ هذا السطر.

*** ملاحظة مهمة :**

١ ــ الأمر C والأمر M يجب أن يقرن بالأمر A أو الأمر B .

٢ ــ لا يمكن مع أمر M أو C ورود أكثر من أمر A أو أمر B أو الأمرين A و B معاً .

```
EDIT — STD4253. TEST. COBOL (PROGA) — 01.20 ———————— COLUMNS 001 072
COMMAND INTPUT ===>                                    SCROLL ===>    HALF
003200            PERFORM READ-WRITE THRU EXT
003300            UNTIL END-OF-JOB = 'YES'.
003400            CLOSE CARD-F PRINT-F.
003500            STOP RUN.
003600    *
003700        READ-WRITE.
003800            READ CARD-F AT END MOVE 'YES' TO
003900            END-OF-JOB GO TO EXT.
004000    *
R04100            MOVE CARD-REC TO PRINT-LINE.
004200            WRITE PRINT -LINE AFTER 1 LINE.
004300    *
004400       EXT.
004500           EXIT.
******      ***************** BOTTOM OF DATA *****************
```

شكل رقم (١١ــ٢١)

```
EDIT — STD4253. TEST. COBOL (PROGA) — 01.20 ———————— COLUMNS 001 072
COMMAND INTPUT ===>                                   SCROLL ===>   HALF
003200            PERFORM READ-WRITE THRU EXT
003300            UNTIL END-OF-JOB = 'YES'.
003400            CLOSE CARD-F PRINT-F-
003500            STOP RUN.
003600    *
003700        READ-WRITE.
003800            READ CARD-F AT END MOVE 'YES' TO
003900            END-OF-JOB GO TO EXT.
004000    *
004100            MOVE CARD-REC TO PRINT-LINE.
004110            MOVE CARD-REC TO PRINT-LINE.
004200            WRITE PRINT-LINE AFTER 1LINE.
004300    *
004400      EXT.
004500           EXIT.
******      **************** BOTTOM OF DATA ****************
```

شكل رقم (١١ــ٢٢)

```
EDIT — STD4253. TEST. COBOL (PROGA) — 01.20 ———————— COLUMNS 001 072
COMMAND INTPUT ===>                                SCROLL ===>    HALF
003200          PERFORM READ-WRITE THRU EXT
003300          UNTIL END-OF-JOB = 'YES'.
003400          CLOSE CARE-F PRINT-F.
003500          STOP RUN.
003600   *
B3700         READ-WRITE.
003800          READ CARD-F AT END MOVE 'YES' TO
003900          END-OF-JOB GO TO EXT.
MM4000   *
004100          MOVE CARD-REC TO PRINT-LINE.
004200          WRITE PRINT-LINE AFTER 1 LINE.
MM4300   *
004400        EXT.
004500          EXIT.
******     ***************** BOTTOM OF DATA *****************
```

شكل رقم (١١ـ٢٣)

```
EDIT — STD4253. TEST. COBOL (PROGA) — 01.20
COMMAND INTPUT ===>                              SCROLL ===     HALF
003200          PERFORM READ-WRITE THRU EXT
003300          UNTIL END-OF-JOB = 'YES'.
003400          CLOSE CARD-F PRINT-F.
003500          STOP RUN.
003600   *
CC3610   *
003620          MOVE CARD-REC TO PRINT-LINE.
003630          WRITE PRINT-LINE AFTER 1 LINE.
CC3640   *
003700       READ-WRITE.
003800          READ CARD-F AT END MOVE 'YES' TO
A03900           END-OF-JOB GO TO EXT.
004400       EXT.
004500          EXIT.
******    ***************** BOTTOM OF DATA *****************
```

شكل رقم (١١ــ٢٤)

ب ــ أوامر عامة GENERAL COMMANDS

هـذه الأ وامـر هيي أوامـر أولـيـة ــ أسـاسية ــ تكتب في مكان إدخال الأ وامر في السطر الثاني من شاشة منقح النصوص TEXT EDITOR .

والشكل العام لها : == COMMAND INPUT > Command[operand] ==

وسنستعرض منها ما يلي :

RESET	LOCATE
SAVE	NUMBER
SUBMIT	RENUM
CANCEL	UNNUM

١ ــ أمر الوصول لسطر معين LOCATING LINES

يستخدم هذا الأمر من أجل الوصول لسطر معين في البرنامج ــ الملف ــ والشكل العام له :

COMMAND INPUT > LOCATE line - number

ويمكن اختصار LOCATE إلى LOC أو L

انظر الشكل رقم (١١ــ٢٥)

٢ ــ توليد وضبط الأ رقام المتسلسلة :

إن منقح النصوص TEXT EDITOR يولد آلياً أرقاماً متسلسلة بالنسبة للأسطر الجـديـدة المضافة بأمر INSERT أو أمر COPY أو يعاد ترقيم البرنامج كله عندما يحفظ على وسط التخزين الثانوي .

```
EDIT — STD4253. TEST. COBOL (PROGA) — 01.20 ————————— COLUMNS 001 072
COMMAND INTPUT ===>L 32                                  SCROLL ===>    HALF
003200            PERFORM READ-WRITE THRU EXT
003300            UNTIL END-OF-JOB   = 'YES'.
003400            CLOSE CARD-F PRINT-F.
003500            STOP RUN.
003600     *
003640     *
003700        READ-WRITE.
003800            READ CARD-F AT END MOVE 'YES' TO
003900            END-OF-JOB GO TO EXT.
004000     *
004100            MOVE CARD-REC TO PRINT-LINE.
004200            WRITE PRINT-LINE AFTER 1 LINE.
004300     *
004400      EXT.
004500            EXIT.
******      ***************** BOTTOM OF DATA *****************
```

شكل رقم (١١ــ٢٥)

صيغة الأرقام المتسلسلة :

تتولد الأرقام المتسلسلة آلياً بالنسبة لبرنامج كوبول ــ تشكل هذه الأرقام الخانات الستة الأولى من كل سطر كوبول تبدأ عادة بالرقم ١٠٠ وتزداد بمعدل ١٠٠ لكل سطر تالي وعند إضافة أسطر جديدة بأمر INSERT تزداد بمعدل ١٠ لتحافظ على التسلسل المنطقي للبرنامج.

في الواقع هناك تسلسل آخر خاص بمنقح النصوص وهو STANDARD الترقيم القياسي. وهو غالباً ما يطابق ترقيم الكوبول. ويحتل الخانات الست الأولى من سطر منقح النصوص.

ضبط تسلسل الترقيم :

يمكن ضبط تسلسل الترقيم بأمر NUMBER والشكل العام للأمر.

COMMAND INPUT > NUMBER ON/OFF [COBOL] [STD]

* ويمكن اختصار الأمر NUMBER إلى NUMB

والتعبير COBOL إلى COB

أما التعبير STD فيعني STANDARD

عندما يكون الأمر NUMBER على الوضع ON أي في حالة عمل، فان هذا الأمر يعمل تحقق Verify لأسطر البرنامج بحيث يتأكد من أن تسلسل الأسطر هو تصاعدي، وكذلك فان هذا الأمر اذا وضع على الوضع ON فانه يعيد ترقيم الأسطر غير المرقمة أو الخارجة عن التسلسل العام، لكنه لا يعدّل تسلسل أرقام الأسطر الموجودة أصلاً.

اما عبارة COBOL فتعني أن الترقيم والتسلسل يتبع قواعد لغة كوبول. أما عبارة STD فتعني أن الترقيم يتبع الترقيم القياسي العام ويمكن جمع التسلسلين معاً.

و يلغى عمل هذا الأمر بكتابة العبارة OFF

> NUMBER OFF

وهناك ثلاثة أوامر تتبع في عملها للأمر NUMBER وهي :

- أمر إعادة الترقيم RENUM
- أمر إلغاء الترقيم UNNUM
- أمر الترقيم الذاتي AUTONUM

ولا تعمل هذه الأوامر إلا اذا كان الأمر NUMBER على الوضع ON

٣ ــ أمر إعادة الترقيم RENUM

والشكل العام له :

COMMAND INPUT > RENUM [STD] [COBOL]

و يعمل هذا الأمر على إعادة ترقيم الأسطر كلها مباشرة، و يبدأ الترقيم بالسطر الأول و بالرقم ١٠٠ و يزداد بمعدل ١٠٠ لكل سطر تالي.

٤ ــ أمر إلغاء الترقيم UNNUM

و يعمل هذا الأمر على الغاء كل حقول الترقيم و يضع مكانها فراغاً BLANKS والشكل العام له :

COMMAND INPUT > UNNUM

٥ ــ أمر الترقيم الذاتي AUTONUM

يعمل هذا الأمر على إعادة ترقيم البرنامج ــ الملف ــ كل مرّة قبل أن يحفظه على وسط التخزين الثانوي، و يلغى عمله بكتابة التعبير OFF والشكل العام.

COMMAND INPUT > AUTONUM OFF/ON

٦ ـ أمر الطباعة PRINT

يمكن إرسال البرنامج آلياً لطباعته عند الخروج من البرنامج بمفتاح التحكم PF15 وذلك بوضع أمر الطباعة على وضع العمل ON ، والشكل العام للأمر:

COMMAND INPUT > PRINT ON/OFF

لكن يفضل وضعه على الحالة OFF لأنه من المربك أن نطبع كل مرّة نطلبه فيها للتعديل .

٧ ـ أمر RESET

يمكن أن يظهر على البرنامج المطلوب في TEXT EDITOR وسائل تحذير أو إعلام مختلفة أو عرض لأوضاع بعض الأوامر، أو العبارات الناتجة عن استخدام بعض الأوامر مثل أمر COLS أو CHANGE مما يؤدي أحياناً إلى إرباك المبرمج من كثرة الرسائل، ونستخدم أمر RESET حتى نلغي كل الرسائل والعبارات التي لا تتبع لهيكل البرنامج والشكل العام للأمر:

> RESET

ويمكن اختصار الأمر RESET بالأحرف RES

٨ ـ أمر SUBMIT للتنفيذ :

يستخدم هذا الأمر لارسال النص المكتوب للتنفيذ EXECUTE في المكان المخصص لذلك و يسمى BACKGROUND ، والنص المكتوب غالباً ما يحتوي على عبارات لغة التحكم بالعمل JCL . والشكل العام للأمر:

> SUBMIT

ويختصر بالأحروف SUB

٩ ــ أمر الحفظ SAVE

مـن المعروف أن مفتـاح التحكـم PF15 ينـهي عمل منقح النـصوص TEXT EDITOR ويحفظ البرنامج المكتوب بنفس الوقت، ولكن عندما يريد المبرمج حفظ برنامجه على وسط التخزين الثانوي دون أن ينهي العمل على البرنامج، فانه يستخدم أمر SAVE لحفظ البرنامج مع الابقاء على الشاشة كما هي عليه، أي يستطيع بعد تنفيذ هذا الأمر أن يستمر بالعمل على نفس البرنامج الشكل العام للأمر:

COMMAND INPUT > SAVE

و ينـصح بعمـل SAVE كلما انتهينا من كتابة صفحة أو كلما أدخلنا تعديلات كبيرة على البرنامج، لأنه في حالة وقوع خطأ في الحاسب وهو ما يسمى عادة CRASH فانه من الممكن أن يخرب التعديلات الحديثة أو أن يلغي برنامجاً جديداً لم يعمل له حفظ SAVE قبل ذلك.

١٠ ــ أمر الالغاء CANCEL

يمكن للمبرمج أن يصرف النظر عن التعديلات التي أدخلها على برنامجه لسبب أو لآخر، لذا فان أمر CANCEL يلغي آخر تعديلات أدخلت على البرنامج قبل أن يعمل له SAVE أو قبل أن يخرج من البرنامج بـ PF15.

والشكل العام للأمر:

COMMAND INPUT > CANCEL

جـ ــ أمر البحث عن حرف أو سلسلة أحرف، أمر التبديل:

FINDING AND CHANGING CHARCTER STRING

وهذان الأمران هما: أمر البحث FIND ويختصر بالحرف F

أمر التبديل CHANGE ويختصر بالحرف C

و يعتبر هاذان الأمران من الأ وامر الأ ولية Primary Commands ، لذا يكتب في السطر الثاني من شاشة منقح النصوص، مقابل عبارة COMMAND INPUT

١ ـــ أمر FIND

يستخدم هذا الأمر لايجاد حرف أو سلسلة أحرف مكررة مرّة أو عدّة مرات في البرنامج ـــ الملف ـــ

والشكل العام للأمر هو:

COMMAND INPUT > FIND string - 1 [ALL]

العبارة ALL تفيد بالبحث عن السلسلة في كل البرنامج ابتداءاً من السطر الأ ول حتى السطر الأخير. و يعطي عدد المرات المكررة في الزاو ية اليمنى العليا من الشاشة.

أما عندما تحذف عبارة ALL فانه يبدأ بالبحث عن أول تسلسل للأحرف المطلو بة مباشرة بعد الموقع الذي هو فيه، فاذا كان بالسطر ٥٠ يبدأ البحث من السطر ٥٠ إلى نهاية البرنامج ـــ الملف ـــ وكلما صادف هذه السلسلة المطلو بة أظهرها بلون غامق وقد وضعت الاشارة CURSOR تحتها وننتقل من سلسلة إلى سلسلة أخرى بضغط مفتاح PF17 لنرى نفس السلسلة في مكان أخر من البرنامج، وعند انتهاء البرنامج ـــ الملف ـــ تظهر لدينا رسالة 'Bottom of Data Reached' في الزاو ية اليمنى العليا من الشاشة. وعندما نضغط PF17 مرّة ثانية تبدأ البحث من جديد على نفس السلسلة من أول البرنامج.

ويجب أن نراعي أنه إذا كانت السلسلة تحتوي على فراغاً SPACE أو فاصلة (,) COMMA ، أو كانت أحد عبارات الأ وامر فيجب وضعها بين فاصلتين علو يتين (') APOSTROPHE أو بين فاصلتين علو يتين مكررتين (") QUOTES أمثلة :

COMMAND INPUT > F XYZ سلسلة بسيطة

سلسلة تحتوي على فراغ COMMAND INPUT > F 'TO WORKI'

سلسلة تحتوي على عبارة COMMAND INPUT > F 'ALL'

مستخدمة في الأ وامر

في حال عدم وجود السلسلة String المطلوب البحث عنها في الملف كله تظهر رسالة 'Not Found' في الزاو ية العليا اليمنى من الشاشة.

٢ ـ أمر CHANGE التبديل :

يستخدم هذا الأمر لايجاد حرف أو سلسلة أحرف وتبديلها بسلسلة أحرف أخرى. والشكل العام للأمر:

COMMAND INPUT > CHANGE String - 1 String - 2 [ALL]

أي بدل السلسلة ١ String - 1 بـ السلسلة ٢ String - 2

ويمكن اختصار الأمر CHANGE إلى الحرف C*

إن عبارة All تفيد بأن يغير السلسلة ١ بـ السلسلة ٢ في كل البرنامج أينما وجدت هذه السلسلة.

مثـــال :

COMMAND INPUT > C WRK1 WRK2 ALL

أي بدل السلسلة من الأحرف WRK 1 بـ السلسلة WRK 2 في كل البرنامج.

يمكن حذف عبارة ALL لأن استخدامها دقيق فهي قد تبدل سلسلة غير مطلوب تبديلها بخطأ من المبرمج.

و يفضل استخدام الأمر CHANGE مع الأمر FIND بحيث يضغط PF17 من أجل ايجاد السلسلة المطلوب تبديلها وعند التأكد من ذلك يضغط مفتاح PF18 من أجل تنفيذ هذا التبديل، وهكذا...

(*) يجب الانتباه إلى أن الحرف C هو اختصار أمر CHANGE وهو أمر أولي، أما الأمر C هو طباعة سطر أمر يكتب على السطر Line Command

أما السلسلة String 1 أو String 2 فتوضع بين فاصلتين علويتين إذا احتويتا أي فراغ أو فاصلة يرجع لأمر F ـ .

عند تبديل أي سلسلة بسلسلة أخرى تظهر اقصى يسار السطر الذي حصل فيه التبديل عبارة 〈 CHG == أي تم التبديل.

ونذكر أنه يمكن تبديل سلسلة بسلسلة أحرف أطول منها أو أقصر. وفي حالة الأطول نجد أن الأمر يزيح الأحرف ليدخل الأحرف الزائدة، وفي حال عدم وجود فراغ مناسب يستوعب الأحرف الزائدة تظهر عبارة 〈 ERR == في أقصى يسار السطر الجاري العمل عليه.

يمكن إزالة عبارة 〈 CHG == أو 〈ERR == بتنفيذ أمر RESET كما مر معنا سابقاً.

كما ويمكن أن يأخذ الأمران FIND و CHANGE صيغا معقدة أخرى، لكن يكتفى بالشكل الحالي الذي يفي بالغرض.

انظر الشكل رقم (١١ـ٢٦) من أجل ملاحظة عمل أمري F و C. وكذلك الشكل رقم (١١ـ٢٧).

د ـ أوامر متقدمة :

وسنستعرض منها :

ـ أوامر سطرية ـ منها :

أمر الازاحة لليسار COLUMNS LEFT ')'

أمر الازاحة لليمين COLUMNS RIGHT '('

— أوامر أساسية :

PROFILE	COPY
HEX	MOVE
	CREATE

١ ــ أمر الازاحة :

يستخدم هذا الأمر اعتباراً من السطر المطلوب بهدف إزاحة الأعمدة لليمين أو اليسار بقدر موضع أو أكثر وتستخدم لذلك الرموز التالية :

SHIFT RIGHT (الازاحة لليمين

SHIFT LEFT) الازاحة لليسار

يدخل هذا الأمر ــ الرمز ــ في المكان المخصص لأوامر الأسطر و يسمى Line Command Area وهو أقصى يسار السطر، ثم يضغط مفتاح ENTER فتتم الازاحة لليمين أو لليسار على حسب الرمز ــ الأمر ــ ، أما إذا أردنا الازاحة أكثر من موقع واحد لليسار نكتب n) بحيث ترمز n إلى عدد الأعمدة المطلوب إزاحتها والصيغة العامة

(n

) n

كما ويمكن إزاحة مجموعة أسطر متتالية BLOCK OF LINES ــ أو برنامج كامل ــ لليمين أو لليسار باستخدام نفس الرموز و بالشكل التالي :

لو فرضنا أننا نريد إزاحة ٢٠ سطر لليسار بمقدار ٨ مواقع فاننا نكتب في السطر الأول من المجموعة الرمز 8)) وفي السطر الأخير من المجموعة)) ــ بدون رقم ــ ونضغط مفتاح ENTER لتتم الازاحة لليسار بمعدل ٨ مواقع 8 POSITIONS .

وتستخدم نفس الخطوات من أجل الازاحة لليمين لكن باستخدام الرمز ((.

انظر الشكل رقم ١١ــ٢٨ ، ١١ــ٢٩ ، ١١ــ٣٠ ، ١١ــ٣١ .

```
EDIT _ STD4253. TEST. COBOL (PROGA) — 01.20                COLUMNS 001 072
COMMAND INTPUT ==⇒C CARD-F CARDF                     SCROLL ===>     HALF
******      ***************** TOP OF DATA * ******************
000100          IDENTIFICATION DIVISION.
000200          PROGRAM-ID. PROGA.
000300          *   THIS PROGRAM PRINTS IPA —
000400          *   NAME AND ADDRESS
000500          *
000600          ENVIRONMENT DIVISION.
000700          CONFIGURATION SECTION.
000800          SOURCE-COMPUTER. IBM-370.
000900          OBJECT-COMPUTER. IBM-370.
001000          *
001100          INPUT-OUTPUT SECTION.
001200          FILE-CONTROL.
001300              SELECT CARD-F ASSIGN TO UT-S-CARD.
001400              SELECT PRINT-F ASSIGN TO UT-PRFL.
001500          *
001600          DATA DIVISION.
001700          FILE SECTION.
001800          FD CARD-F RECORD CONTAINS 80 CHARACTERS
001900              LABEL RECORD IS OMITTED.
002000          01 CARD-REC PIC X (80).
002100          *
```

شكل رقم (١١ـ٢٦)

```
EDIT — STD4253. TEST. COBOL (PROGA) — 01.20_____ CHARS 'CARD-F' CHANGED
COMMAND INTPUT ===>                                 SCROLL ===>    HALF
******      ****************** TOP OF DATA * ******************
000100        IDENTIFICATION DIVISION.
000200        PROGRAM-ID. PROGA.
000300        *        THIS PROGRAM PRINTS IPA —
000400        *        NAME AND ADDRESS
000500        *
000600        ENVIRONMENT DIVISION.
000700        CONFIGURATION SECTION.
000800        SOURCE-COMPUTER. IBM-370.
000900        OBJECT-COMPUTER. IBM-370.
001000        *
001100        INPUT-OUTPUT SECTION.
001200        FILE-CONTROL.
001300            SELECT CARDF ASSIGN TO UT-S-CARD.
001400            SELECT PRINT-F ASSIGN TO UT-PRFL.
001500        *
001600        DATA DIVISION.
001700        FILE SECTION.
001800        FD CARDF RECORD CONTAINS 80 CHARACTERS
001900            LABEL RECORD IS OMITTED.
002000        01 CARD-REC PIC X (80).
002100        *
```

شكل رقم (١١ ــ ٢٧)
بعد تنفيذ أمر التغيير

٢ ــ أمر COPY و MOVE

يستخدم هذان الأمران كأوامر أولية أساسية PRIMARY COMMANDS، لذا يكتب في السطر الثاني مقابل عبارة.

.... COMMAND INPUT

و يستخدم هذان الأمران لدمج برنامج ــ ملف ــ جزئي مع برنامج ــ ملف ــ أساسي MERGE DATA ، ونعيد إلى الأذهان الفرق بين عبارتي COPY و MOVE إذ أن الأولى تنسخ نسخة لمكان آخر و يظل الأصل كما هو، بين الثانية ــ MOVE ــ يتم نقل المحتوى كله من مكان لآخر دون الاحتفاظ بأي أصل.

```
EDIT — STD4253. TEST. COBOL (PROGA) — 01.20____SCREEN IMAGE IS PRINTED
COMMAND INTPUT ===>                                    SCROLL ===>     HALF
******      ****************** TOP OF DATA * ******************
000100          IDENTIFICATION DIVISION.
000200          PROGRAM-ID. PROGA.
000300          *         THIS PROGRAM PRINTS IPA_
000400          *         NAME AND ADDRESS
000500          *
((200           ENVIRONMENT DIVISION.
000700          CONFIGURATION SECTION.
000800          SOURCE-COMPUTER. IBM-370.
((0900          OBJECT-COMPUTER. IBM-370.
001000          *
001100          INPUT-OUTPUT SECTION.
001200          FILE-CONTROL.
001300
001400              SELECT PRINT-F ASSIGN TO UT-PRFL.
001500          *
001600          DATA DIVISION.
001700          FILE SECTION.
001800          FD CARD-F RECORD CONTAINS 80 CHARACTERS
001900              LABEL RECORD IS OMITTED.
002000          01 CARD-REC PIC X (80).
002100          *
```

شكل رقم (١١ــ٢٨)

```
EDIT — STD4253. TEST. COBOL (PROGA) — 01.20 ———————— COLUMNS 001 072
COMMAND INTPUT ===>                                SCROLL ===>   HALF
******      ****************** TOP OF DATA ********************
000100           INDENTIFICATION DIVISION
000200           PROGRAM-ID. PROGA.
000300        *            THIS PROGRAM PRINTS IPA—
000400        *            NAME AND ADDRESS
000500        *
000600       ENVIRONMENT DIVISION.
000700       CONFIGURATION SECTION.
000800       SOURCE-COMPUTER. IBM-370.
000900       OBJECT-COMPUTER. IBM-370.
001000        *
001100           INPUT-OUTPUT SECTION.
001200           FILE-CONTROL.
001300               SELECT CARD-F ASSIGN TO UT-S-CARD.
001400               SELECT PRINT-F ASSIGN TO UT-PRFL.
001500        *
001600           DATA DIVISION.
001700           FILE SECTION.
001800           FD CARD-F RECORD CONTAINS 80 CHARACTERS
001900               LABEL RECORD IS OMITTED.
002000           01 CARD-REC PIC X (80).
002100        *
```

شكل رقم (١١ــ٢٩)

```
EDIT — STD4253. TEST. COBOL (PROGA) — 01.20 ____SCREEN IMAGE IS PRINTED
COMMAND INTPUT ===>                              SCROLL ===>    HALF
******    ****************** TOP OF DATA ********************
000100          IDENTIFICATION DIVISION.
000200          PROGRAM-ID. PROGA.
000300          *          THIS PROGRAM PRINTS IPA—
000400          *       NAME AND ADDRESS
000500          *
))0600              ENVIRONMENT DIVISION.
000700              CONFIGURATION SECTION.
000800              SOURCE-COMPUTER. IBM-370.
))0900              OBJECT-COMPUTER. IBM-370.
001000          *
001100          INPUT-OUTPUT SECTION.
001200          FILE-CONTROL.
001300            SELECT CARD-F ASSIGN TO UT-S-CARD.
001400            SELECT PRINT-F ASSIGN TO UT-PRFL.
001500          *
001600          DATA DIVISION.
001700          FILE SECTION.
001800          FD CARD-F RECORD CONTAINS 80 CHARACTERS
001900            LABEL RECORD IS OMITTED.
002000          01 CARD-REC PIC X (80).
002100          *
```

شكل رقم (١١ـ٣٠)

```
EDIT _ STD4253. TEST. COBOL (PROGA) _ 01.20__________COLUMNS 001  072
COMMAND INTPUT  ===>                              SCROLL  ===>    HALF
******    ****************** TOP OF DATA ********************
000100         IDENTIFICATION DIVISION.
000200         PROGRAM-ID. PROGA.
000300         *   THIS PROGRAM PRINTS IPA_
000400         *   NAME AND ADDRESS
000500         *
000600         ENVIRONMENT DIVISION.
000700         CONFIGURATION SECTION.
000800         SOURCE-COMPUTER. IBM-370.
000900         OBJECT-COMPUTER. IBM-370.
001000         *
001100         INPUT-OUTPUT SECTION.
001200         FILE-CONTROL.
001300             SELECT CARD-F ASSIGN TO UT-S-CARD.
001400             SELECT PRINT-F ASSIGN TO UT-PRFL.
001500         *
001600         DATA DIVISION.
001700         FILE SECTION.
001800         FD CARD-F RECORD CONTAINS 80 CHARACTERS
001900              LABEL RECORD IS OMITTED.
002000         01 CARD-REC PIC X (80).
002100         *
```

شكل رقم (١١ـ٣١)

وتتم عملية الدمج حسب الخطوات التالية :

١ ــ يطلب البرنامج الأصلي المراد إضافة برنامج جزئي ــ ملف جزئي ــ إليه بالخيار EDIT 2

٢ ــ يحدد الموقع ضمن البرنامج الأصلي الذي سيدخل الجزء الاضافي إليه وذلك بوضع الأمر A (After) أو B (Before) في السطر المناسب و بالمكان المناسب Line Commands Area

٣ ــ يكتب في السطر الثاني مكان الأ وامر الأساسية الأمر COPY أو MOVE حسب احتياجات العمل، وأمامه اسم الملف الذي سيدخل على البرنامج الأساسي. وكما هو مبين :

COMMAND > INPUT COPY Filename

↓

يكتب هنا اسم الملف
الجزئي المطلوب ادخاله

٤ ــ اضغط مفتاح ENTER لتنفيذ الأمر.

٣ ــ أمر REPLACE, CREATE

وهما من الأ وامر الأساسية Primary Commands والشكل العام لهما :

COMMAND INPUT > { CREATE / REPLACE } Filename

و يستخدم هذان الأمران عندما نريد فصل أو تجزئة برنامج ــ ملف ــ إلى ملف فرعي آخر.

مثال : لو أردنا فصل جزء من برنامج إلى برنامج فرعي نتبع ما يلي :

١ ــ نطلب البرنامج الأساسي ــ الأصلي ــ بالخيار EDIT 2

٢ ــ نحدد الأسطر التي نريد فصلها نهائياً بأمر MM أو إذا أردنا نسخ صورة منها نستخدم الأمر CC وهما أمران على مستوى السطر Line Command.

٣ ــ نكتب في المكان المخصص لكتابة الأوامر الأساسية الأمر CREATE كما هو واضح.

COMMAND INPUT > CREATE Filename

↓

اسم الملف الجديد

الذي سيحتوي على الأسطر المنقولة

٤ ــ نضغط مفتاح ENTER من أجل تنفيذ هذا الأمر.

بعد تنفيذ هذا الأمر يتشكل لدينا ملف جديد يحتوي على الأسطر المنقولة أو المنسوخة من الملف الأساسي.

File name وهو اسم الملف فيجب ان يكون جديداً غير مستخدم مسابقاً NOT EXIST عند استخدام أمر CREATE ولا يشترط ذلك عند استخدام أمر REPLACE لأنه ولو كان هناك ملف ــ برنامج ــ بنفس الاسم فان أمر REPLACE سوف يلغي القديم ويحل محله الجديد.

٥ ــ استخدام أمر PROFILE إطار الملف.

يمكن التعرف على حالة بعض الأوامر فيما إذا كانت في حالة عمل ON أو عدم العمل OFF وذلك بكتابة أمر PROFILE كتعليمة أساسية فتظهر هذه المعلومات على الأسطر الثلاثة الأولى من الشاشة الشكل العام لها :

> PROFILE

انظر الشكل رقم (١١ ــ ٣٢)

```
EDIT — STD4253. TEST. COBOL (PROGA) — 01.20____________COLUMNS 001 072
COMMAND INTPUT ==>PROFILE                          SCROLL ==>    HALF
******     ******************** TOP OF DATA ********************
= PROF >  ....COBOL (FIXED - 80) .... RECOVERY ON ... NUMBER ON STD ...........
= PROF >  .... CAPS ON .... HEX OFF .... NULLS OFF .... TABS OFF ........................
= PROF >  .... AUTONUM OFF .... PRINT OFF .... STATS ON ...................................
000100          IDENTIFICATION DIVISION.
000200          PROGRAM-ID. PROGA.
000300          *          THIS PROGRAM PRINTS IPA—
000400          *          NAME AND ADDRESS
000500          *
000600          ENVIRONMENT DIVISION.
000700          CONFIGURATION SECTION.
000800          SOURCE-COMPUTER. IBM-370.
000900          OBJECT-COMPUTER. IBM-370.
001000          *
001100          INPUT-OUTPUT SECTION.
001200          FILE-CONTROL.
001300              SELECT CARD-F ASSIGN TO UT-S-CARD.
001400              SELECT PRINT-F ASSIGN TO UT-PRFL.
001500          *
001600          DATA DIVISION.
001700          FILE SECTION.
001800          FD CARD-F RECORD CONTAINS 80 CHARACTERS
```

شكل رقم (١١ ــ ٣٢)

٦ ـ استخدام أمر HEX

يستخدم هذا الأمر لتحويل الأحرف المكتوبة من الصيغة العامة STANDARD FORM إلى الصيغة السداسية عشر HEXADECIMAL والشكل العام للأمر:

> HEX ON / OFF

وعند وضع هذا الأمر على ON فان كل سطر من سطر البيانات سوف يترجم إلى سطرين بالصيغة السداسية عشر HEXADECIMAL

ويمكن تبديل أي حرف بالصيغة السداسية عشر إلى حرف آخر بنفس الصيغة. ويمكن الغاء هذه الحالة بكتابة HEX OFF

فتعود الكتابة إلى الشكل العام العادي.

انظر الشكل رقم (١١ـ٣٣) والشكل رقم (١١ـ٣٤).

```
EDIT — STD4253. TEST. COBOL (PROGA) — 01.20 ———————— COLUMNS 001  072
COMMAND INTPUT  ===> HEX ON                              SCROLL  ===>    HALF
******     ********************  TOP OF DATA  *******************
_____________________________________________________________________
000100              IDENTIFICATION DIVISION.
        4444444CCCECCCCCECDD4CCECECECDD 444444444444444444444444444444
        0000000945539693139650495929 65B 0000000000000000000000000000000000000
_____________________________________________________________________
000200          PROGRAM - ID. PROGA.
        4444444DDDCDCD6CC44444DDDCC 444444444444444444444444444444444444
        0000000796791409 4B000079671B 00000000000000000000000000000000000000000
_____________________________________________________________________
000300       *    THIS PROGRAM PRINTS IPA—
        44444454444444444444ECCE4DDD4DDCDEE44CDC6 444444444444444444444
        000000000000000000038920796791407995320097 1D 0000000000000000000000000
_____________________________________________________________________
000400       *    NAME AND ADDRESS
        44444454444444444444DCDC4CDC4CCDC4CCDCDCEE 444444444444444444444
        000000C00000000000051450154015401494 9522 00000000000000000000000000000
_____________________________________________________________________
```

شكل رقم (١١ ــ ٣٣)

```
EDIT _ STD4253. TEST. COBOL (PROGA) _ 01.20 ___________ COLUMNS 001 072
COMMAND INTPUT ===> HEX OFF                     SCROLL ===>    HALF
******      ****************** TOP OF DATA *********************
000100      IDENTIFICATION DIVISION.
000200      PROGRAM-ID. PROGA.
000300      *          THIS PROGRAM PRINTS IPA_
000400      *          NAME AND ADDRESS
000500      *
000600      ENVIRONMENT DIVISION.
000700      CONFIGURATION SECTION.
000800      SOURCE-COMPUTER. IBM-370.
000900      OBJECT-COMPUTER. IBM-370.
001000      *
001100      INPUT-OUTPUT SECTION.
001200      FILE-CONTROL.
001300          SELECT CARD-F ASSIGN TO UT-S-CARD.
001400          SELECT PRINT-F ASSIGN TO UT-PRFL.
001500      *
001600      DATA DIVISION.
001700      FILE SECTION.
001800      FD CARD-F RECORD CONTAINS 80 CHARACTERS
001900      LABEL RECORD IS OMITTED.
002000      01 CARD-REC PIC X (80).
002100      *
```

شكل رقم (١١ ـ ٣٤)

١١ ــ ٤ استخدام نظام الحاسب TI990

١١ - ٤ - ١ استخدام TEXT EDITOR منقح النصوص في جهاز TEXAS INSTRUMENTS

يستخدم منقح النصوص TEXT EDITOR كوسيلة لادخال ملفات، أو لتعديل ملفات قديمة و يستخدم عبر الشاشات TERMINALS

يمكن استدعاء منقح النصوص هذا بتعليمة XE (EXECUTE EDIT)، ويمكن بعدها تعيين اسم الملف إذا كان الملف قديماً، أو ادخال ملف جديد كما سيرد شرحه فيما بعد. و يفترض في هذه الحالة أن المبرمج قد أدخل USER ID والـ PASSCODE الخاصة به.

عندما نريد استخدام TEXT EDITOR من أجل تعديل ملف قديم، ندخل اسم الملف و يظهر الملف على الشاشة ٢٤ سطراً مرقماً من ١ إلى ٢٤ ويجري إضافة أحرف جديدة أو حذف أحرف وإضافة أسطر والغاء أسطر لكن كل ذلك لا يتناول النسخة الأصلية للملف إنما نسخة ثانية تسمى ملف التعديل.

نتنهي من عملية التنقيح أي نخرج من TEXT EDITOR بتعليمة

QE (QUIT EDIT)

ويمكن للمبرمج أن يلغي كل التعديلات التي أدخلها أو أن يثبت هذه التعديلات ويمكن له أن يكتب الملف المعدّل على ملف آخر ويحتفظ بالملف الأساسي بدون تعديل.

١١ - ٤ - برنامج جديد :

كيفية استخدام الشاشات في TEXT EDITOR ؟

يفترض أننا فتحنا الشاشة وذلك بادخال الرمز التعريفي ــ وكلمة السر بعدها نكتب عبارة XE فتظهر لدينا الشاشة التالية بعد ضغط مفتاح RETURN

INITIATE TEXT EDITOR

FILE ACCESS NAME : اسم البرنامج

نضغط RETURN دلالة على اننا نريد إنشاء ملف (برنامج) جديد فتظهر امامنا شاشة نظيفة عليها السطر التالي :

EOF *

وهذا معناه أن الملف فارغ ويحتوي فقط علامة نهاية الملف END OF FILE نضغط مفتاح التحكم F7 وكذلك المفتاح الرمادي غير المعنون وعندها ستنتقل علامة EOF * للسطر الثاني وتبدأ بادخال برنامج سطراً سطراً، وكلما انتهينا من كتابة سطر ـــ سجل RECORD ــ أضغط ENTER ليأخذ السطر التالي آلياً.

ويمكننا خلال وجودنا في TEXT EDITOR استخدام أي أمر من أوامر التنقيح والتي سيرد ذكرها فيما بعد.

بعد الانتهاء من ادخال البرنامج، نضغط مفتاح CMD ونكتب QE فتظهر لدينا الرسالة أو الشاشة التالية :

QUIT EDIT

ABORT ? : NO هل يخرب العمل ؟

هذا معناه إننا إذا أردنا الغاء ما كتبناه نكتب YES و بالتالي سيلغي الحاسب كل شيء كتبناه، اما إذا أجبنا بـ NO فهذا معناه أنه سيحتفظ بما كتبناه سابقاً، بعدها نضغط RETURN لتظهر أمامنا الشاشة التالية :

QUIT EDIT

OUT PUT FILE ACCESS NAME : اسم البرنامج

REPLACE ? : NO

MOD LIST ACCESS NAME :

OUT PUT ACCESS NAME يعني اسم البرنامج المراد تخزينه أي نكتب اسم البرنامج

والسطر الثاني REPLACE ، أي هل يبدل هذه النسخة بالنسخة السابقة ؟

في حالتنا هذه لا يوجد نسخة سابقة ، فلو ادخلنا الجواب YES أو NO (نعم أو لا) لا فرق في ذلك.

MOD LIST ACCESS NAME

و يذكر هنا اسم الملف الذي يحتوي على التعديلات ، وكذلك في حالتنا هذه لا يوجد تعديلات بل أن الملف جديد لذلك نضغط RETURN

بهذا يكون العمل على منقح النصوص انتهى وتم حفظ البرنامج على وسط التخزين الثانوي .

١١ ــ ٤ ــ ٢ ــ تنقيح ملف موجود سابقاً في جهاز TI

إن أي برنامج يدخل للحاسب لابد وأنه يحتاج إلى تنقيح ، أمّا من أخطاء قواعدية أو من أخطاء منطقية ، وأحياناً يتطلب التعديل ، تغيير في المعطيات الأساسية للبرنامج .

فتسلسل عملية طلب الملف ــ البرنامج ــ للتعديل تتم وفقاً للخطوات التالية :

١ ــ نطلب TEXT EDITOR بالأمر XE

٢ ــ تظهر لدينا الشاشة التالية :

INITIATE TEXT EDITOR

FILE ACCESS NAME : اسم البرنامج

فتدخل اسم البرنامج الموجود أصلاً في الحاسب ، فيظهر أمامنا البرنامج على الشاشة اعتباراً من السطر الأول ولغاية السطر ٢٤ وأمام كل سطر يوجد رقم السطر المتسلسل .

يمكننا طلب الصفحات التالية للبرنامج بضغط مفتاح F1 أو يمكننا الرجوع صفحات للوراء بضغط مفتاح F2 كما ويمكننا استخدام الاسهم ⇄ ↑ ↓ لتحريك الاشارة المضيئة CURSOR ووضعها على الأحرف المطلوبة وتعديل هذه الأحرف.

كما يمكننا استخدام مفاتيح DEL CHAR لحذف أحرف معينة أو مفتاح INS CHAR لاضافة أحرف معينة بين سلسلة من الأحرف ضمن السطر الواحد. كما يمكننا حذف السطر وإضافة السطر جديدة للملف ــ البرنامج ــ

عند الانتهاء من التعديل اضغط مفتاح CMD البرتقالي ونكتب QE فتظهر أمامنا الشاشة التالية :

QUIT EDIT

ABORT ? NO

إذا كنا لا نرغب في التعديلات التي عملنا عليها فنكتب YES و بالتالي يرجع الملف ــ البرنامج ــ كما كان أصلاً دون أي تعديل وكأن شيء لم يكن. أما إذا كنا جادين في التعديل فنكتب NO و بالتالي ستظهر لدينا الشاشة التالية :

QUIT EDIT

OUTPUT FILE ACCESS NAME :

REPLACE ? : YES

MOD LIST ACCESS NAME :

عند عبارة OUTPUT FILE ACCESS NAME يجب أن نضع اسم البرنامج الذي كنا نعمل عليه، أو اسم البرنامج ــ ملف ــ جديد نرغب في إنشاءه و بالتالي يكون عندنا برنامجين الأصل والنسخة المعدّلة، لكن في الأحوال العادية نستخدم نفس اسم البرنامج في الأول وفي النهاية

أي عند البداية FILE ACCESS NAME : TEST

وعند النهاية OUTPUT FILE ACCESS NAME : TEST

أما عبارة REPLACE فهي تعني أن النسخة الجديدة المنقحة ستحل محل النسخة الأصلية إذا كان الجواب YES ، أما إذا كان الجواب NO فتعود للبرنامج وتكتبُ QE من جديد وتعين اسم ملف الاخراج جديد :

OUTPUT FILE ACCESS NAME

أما عبارة MODE LIST ACCESS NAME فهو اسم ملف يحوي على الأسطر المعدلة وأصل هذه الأسطر، ويمكن إهمالها هنا واضغط فقط RETURN لتعود للشاشة الأصلية للحاسب.

استخدام بعض أوامر TEXT EDITOR

يمكن تلخيص بعض أوامر منقح النصوص كما يلي :

	الأمـــر
طباعة أسطر COPY LINES	CL
حذف أسطر DELETE LINES	DL
تحريك أسطر MOVE LINES	ML
حذف سلسلة أحرف DELETE STRING	DS
تغيير سلسلة أحرف REPLACE STRING	RS
ايجاد سلسلة أحرف FIND STRING	FS
تغيير مكان الأعمدة MODIFY TAB	MT
البحث عن أسطر SHOW LINES	SL

* كل هذه الأوامر تنفذ بعد ضغط مفتاح CMD

طباعة أسطر CL : COPY LINES

يستخدم هذا الأمر من أجل طباعة ـ نسخ ـ سطر أو أكثر من مكان معين إلى مكان آخر وعند كتابة هذا الأمر تظهر الشاشة التالية :

COPY LINES

START LINE : رقم سطر البداية

END LINE : رقم سطر النهاية

INSERT AFTER LINE : رقم السطر الذي بعده سيتم ادخال الأسطر الجديدة

حذف أسطر DL DELETE LINES

و يستخدم هذا الأمر لحذف سطر أو مجموعة أسطر، وعند كتابة هذا الأمر تظهر لدينا الشاشة التالية :

DELETE LINES

START LINE : رقم سطر البداية المراد حذفه

END LINE : رقم سطر النهاية المراد حذفه

يتم حذف الأسطر من رقم البداية إلى رقم النهاية وما بينهما .

تحريك أسطر ML MOVE LINE

يستخدم هذا الأمر لتحريك سطر أو عدّة أسطر من مكان في الملف لمكان آخر في الملف ـ البرنامج ـ وعند كتابة هذا الأمر تظهر لدينا الشاشة التالية :

MOVE LINES

START LINE : رقم بداية الأسطر المراد تحريكها

END LINE : رقم نهاية الأسطر المراد تحريكها

INSERT AFTER LINE : رقم السطر المراد ادخال الأسطر بعده

حذف سلسلة أحرف DELETE STRING DS

هذا الأمر يحذف سلسلة من الأحرف يعينها المبرمج وعند كتابة هذا الأمر تظهر لديك الشاشة التالية :

DELETE STRING

NUMBER OF OCCURENCES : عدد مرّات تكرار السلسلة المراد حذفها

END COLUMN : عمود البداية

START COLUMN : عمود النهاية

STRING : السلسلة

ايجاد سلسلة أحرف FS FIND STRING

يستخدم هذا الأمر من أجل البحث عن سلسلة أحرف يعينها المبرمج وعند كتابة هذا الأمر تظهر لك الشاشة التالية :

FIND STRING

OCCURENCE NUMBER : رقم السلسلة المراد تعيينها

START COLUMN : رقم عمود البداية

END COLUMN : رقم عمود النهاية

STRING : السلسلة .

تغيير مجموعة أحرف بمجموعة ثانية REPLACE STRING RS

يستخدم هذا الأمر لابدال سلسلة أحرف مسلسلة أحرف ثانية يعينها المبرمج وعند كتابة هذا الأمر تظهر لديك الشاشة التالية :

REPLACE STRING :

عـدد المـرات المـراد NUMBER OF OCCURENCES :

ابـدال السلسلة بسلسلة الثانية

عمود البداية START COLUMN :

عمود النهاية END COLUMN :

السلسلة الأصلية STRING :

السلسلة البديلة. CHANGE :

تعديل أعمدة . MT MODIFY TAB

يستخدم هذا الأمر لتعديل الأعمدة التي ستقفز عنها الاشارة المضيئة CURSOR ، وعند كتابة هذا الأمر تظهر لديك الشاشة التالية :

MODIFY TAB

يدخل رقم الأعمدة التي يريد للاشارة المضيئة المرور عليها و بينها فاصلة (,) TAB COLUMNS :

أظهر سطر رقم SL SHOW LINE

يستخدم هذا الأمر من أجل وضع أو أظهار رقم السطر الموجود في هذا الأمر على الشاشة ، و يمكن وضع الحرف B لاظهار أسطر البداية ، أو الحرف E لاظهار أحرف النهاية أو أي رقم سطر.

وعند كتابة هذا الأمر يظهر لديك الشاشة التالية :

SHOW LINE

رقم السطر LINE :

الفصل الثاني عشر

ترجمة البرنامج وتنفيذه

COMPILE AND EXECUTE COBOL PROGRAM

١٢ ـ ١ لمحة عامة :

سنتحدث في هذا الفصل : أهمية المترجم واستخدامه في نوعين من الأجهزة المتوفرة في المعهد: الأول الحاسب I.B.M.3033 والثاني الحاسب T.I.990 .

ولابد من التذكير بأن المترجم هو من البرامج الأساسية SOFTWARE الموجودة في كل حاسب يستخدم لغة كوبول.

ويختلف المترجم من حاسب لآخر طبقا لنظام التشغيل المستخدم ومساحة الذاكرة الرئيسية المتوفرة ولكن تبقى الاختلافات ليست جوهرية بالنسبة للغة ككل، ونؤكد بنصح المبرمجين بالرجوع إلى مراجع الشركة التي تشرح وتبين امكانيات المترجم المتوفر مع الحاسب.

١٢ ـ ٢ ترجمة وتنفيذ البرنامج باستخدام الحاسب IBM3033 / نظام التشغيل MVS

إن أهمية المترجم ترجع إلى أنه يقوم بترجمة اللغة العليا ـ في حالتنا هذه الكوبول ـ إلى لغة الآلة و يصيغها بصيغة التنفيذ OBJECT

ويمكن عمل ترجمة للبرنامج بعدة طرق منها :

١ ـ استخدام JCL خاص وتنفيذه بأمر SUBMIT

٢ ـ استخدام FOREGROUND أي الخيار 4.2 والكشف عن الناتج

بـ USER-ID. LIST

٣ ـــ اسـتـخـدام BACKGROUND أي الخـيـار 5.2 والـكـشـف عـن الأخـطـاء بالخيار 3.8

وسنستعرض هنا الطريقة رقم ٣ لبساطتها وسهولة التعرف على الأخطاء.

استخدام BACKGROUND الخيار 5.2 لعمل ترجمة COMPILE للبرنامج :

اعـتـبـاراً مـن الـشـاشـة الأسـاسـية رقم ١١ـــ١ نطلب الخيار 5 فتأتينا الشاشة رقم ١٢ـــ١ تعدّل بطاقات العمل JCL كما هو وارد في الشكل.

بـعـدهـا نـكـتـب الخيار 2 من هذه الشاشة مقابل عبارة SELECT OPTION انظر الشاشة رقم ١٢ـــ١.

بـعـد ان نـضـغـط مـفـتـاح ENTER تأتي الشاشة رقم ١٢ـــ٢ نكتب اسم البرنامج المطلوب عمل ترجمة له ونضغط مفتاح ENTER عندها تعود الشاشة رقم ١٢ـــ٣ نضغط PF15 أي ENDKEY حتى ينفذ العمل وإذا أردنا الغاءه كتبنا عبارة CANCEL

نقرأ اسم العمل الذي تم تنفيذه وهو عبارة عن الرقم التعريفي USER-ID الخاص مـضـافـاً إلـيـه حرف (من A إلى Z) أو رقم، وهذا الاسم هو الذي سنطلبه عند التعرف على الأخطاء في الخيار 3.8.

BACKGROUND SELECTION MENU

SELECTION OPTION ===>

1 _ SYSTEM ASSEMBLER

2 _ OS / VS COBOL COMPILER

3 _ FORTRAN IV (G1) COMPILER

4 _ PL / I CHECKOUT COMPILER

5 _ PL / I OPTIMIZING COMPILER

6 _ LINKAGE EDITOR

9 _ PASCAL / VS COMPILER

JOB STATEMENT INFORMATION : (VERIFY BEFORE PROCEEDING)

```
===> // STD,XXXXY* JOB (STD,XXXX),'NNNN',
===> // CLASS = H,MSGCLASS = Y,
===> // MSGLEVEL = (1,1), NOTIFY = STDXXXX
===> // *
===> // *
```

الشاشة رقم (١٢ـ١)

* الحرف Y : يقصد به أي حرف من A إلى Z

```
________________ BACKGROUND COBOL COMPILE ________________
ENTER / VERIFY PARAMETERS BELOW :

PROJECT    ===>   STDXXXX
LIBRARY    ===>   TEST  ===>      ===>      ===>
TYPE       ===>   COBOL
MEMBER     ===>

LIST ID      ===>                 (BLANK FOR HARDCOPY LISTING)
SYSOUT CLASS   ===>   Y           (IF HARDCOPY REQUESTED)

COMPILER OPTIONS :
   TEST    ===>   NOTEST          (TEST OR NOTEST)
   OTHER   ===>   SOURCE
```

شاشة رقم (١٢ــ٢)

```
_______ BACKGROUND SELECTION MENU ____________ JCL GENERATEDA

SELECTION OPTION  ===>

 1 _ SYSTEM ASSEMBLER                4 _ PL / I CHECKOUT COMPILER
 2 _ OS / VS COBOL COMPILER          5 _ PL / I OPTIMIZING COMPILER
 3 _ FORTRAN IV (G1) COMPILER        6 _ LINKAGE EDITOR
                                     9 _ PASCAL / VS COMPILER

ENTER OPTION TO CONTINUE GENERATING JCL
ENTER CANCEL ON OPTION LINE TO EXIT WITHOUT SUMITTING JOB
PRESS END KEY TO SUBMIT JOB

JOB STATEMENT INFORMATION :  (VERIFY BEFORE PROCEEDING)

===>   / / STD,XXXXY*     JOB    (STD,XXXX),'NNNN',
===>   / / CLASS = H,MSGCLASS = Y,
===>   / / MSGLEVEL = (1,1), NOTIFY = STD XXXX
===>   / / *
===>   / / *
```

الشاشة رقم (١٢ ــ ٣)

* الحرف Y : أي حرف من A إلى Z

الكشف عن الأخطاء الناتجة من عمل COMPILE في الخيار 5.2 :

بعد الانتهاء من مرحلة الترجمة لابد وان نجد طريقة لاستعراض ناتج الترجمة، إذ من النادر جداً كتابة برنامج دون أخطاء قواعدية أو منطقية.

و يتم الكشف عن ناتج عملية الترجمة COMPILE والذي استخدمنا فيها الخيار 5.2 بكتابة الخيار 3.8 اعتباراً من الشاشة الأساسية فتظهر الشاشة رقم ١٢ ـــ٦ نضع الاسم التعريفي أي اسم العمل JOB-ID والذي أخذناه مسبقاً من الخيار 5.2 مقابل العبارة JOB NAME

ننتظر قليلاً ونضغط مفتاح ENTER فتظهر لنا عبارات JCL بعدها نضغط PF20 لنستعرض البرنامج وفي نهاية البرنامج نجد الأخطاء ورقم الأسطر الموجودة فيها.

لا يمكن تصحيح هذه الأخطاء هنا في الخيار 3.8 بل يفضل طباعة هذا العمل على الطابعة إذا كانت الأخطاء كثيرة أو عمل SPLIT للشاشة أي فصل الشاشة باستخدام PF14 وطلب نفس البرنامج بالخيار EDIT 2 وتصحيح هذه الأخطاء و يستعان بالمفتاح PF21 للانتقال من شاشة إلى أخرى.

من الممكن طبعاً وهو الاسهل الخروج من الخيار 3.8 بعد طباعته، أو بعد حفظ الأخطاء أو تدوينها على الورق، ثم الدخول إلى برنامج EDIT وطلب البرنامج المراد العمل عليه.

يفضل عند الانتهاء من استعراض أي عمل حذف ذلك العمل بوضع الحرف D في SELECT OPTION وذلك لعدم تراكم الأعمال مما يشغل حيز كبير على القرص وكذلك تتكرر أسماء الأعمال مما يجعل استعراضها صعباً.

UTILITY SELECTION MENU

```
SELECT OPTION  ===>

1  LIBRARY     _ LIBRARY UTILITY :
                    PRINT INDEX LISTING OR ENTIRE DATASET
                    PRINT, RENAME, DELETE, OR BROWSE MEMBERS
                    COMPRESS DATASET
2  DATASET     _ DATASET UTILITY :
                    DISPLAY DATASET INFORMATION
                    ALLOCATE, RENAME, OR DELETE ENTIRE DATASET
                    CATALOG OR UNCATALOG DATASET
3  MOVE COPY   _ MOVE OR COPY MEMBERS OR DATASETS
4  CATALOG     _ CATALOG MANAGEMENT :
                    DISPLAY OR PRINT CATALOG ENTRIES
                    INITIALIZE OR DELETE USER CATALOG ALIAS
5  RESET       _ RESET STATISTICS FOR MEMBERS OF SPF LIBRARY
6  HARDCOPY    _ INITIATE HARDCOPY OUTPUT
7  VTOC        _ DISPLAY OR PRINT VTOC ENTRIES FOR A DASD VOLUME
8  OUTLIST     _ DISPLAY, DELETE, OR PRINT HELD JOB OUTPUT
9  SCRIPT/VS   _ FORMAT, DISPLAY, AND OPTIONALLY PRINT SCRIPT TEXT
```

شكل رقم (١٢ـ٤)

BACKGROUND LINKAGE EDIT

ENTER/VERIFY PARAMETERS BELOW :

PROJECT ===> STD XXXX

LIBRARY ===> TEST ===> ===> ===>

TYPE ===> OBJ

MEMBER ===>

اسم البرنامج

LIST ID ===> (BLANK FOR HARDCOPY LISTING)

SYSOUT CLASS ===> Y (IF HARDCOPY REQUESTED)

LINKAGE EDITOR OPTIONS :

TERM ===> (TERM OR BLANK)

OTHER ===> LET, LIST, MAP

COMPILER SUBROUNTINE LIBRARIES : (INAPOSTROPHES)

===>

===>

شكل رقم (١٢ـ٥)

OUTLIST UTILITY

SELECT OPTION ===>

L — LIST JOB NAMES/ID'S VIA THE TSO STATUS COMMAND لاظهار الأعمال قيد التنفيذ

D — DELETE JOB OUTPUT FROM SYSOUT HOLD QUEUE لحذف نتيجة عمل منفذ

P — PRINT JOB OUTPUT AND DELETE FROM SYSOUT HOLD QUEUE

طباعة نتيجة عمل على الطابعة المركزية وحذفه

R — REQUEUE JOB OUTPUT TO A NEW OUTPUT CLASS طباعة نتيجة عمل على الطابعة المحلية

BLANK — DISPLAY JOB OUTPUT لاظهار نتيجة عمل على الشاشة

FOR JOB TO BE SELECTED :

JOB NAME ===> STDXXXXY

CLASS ===>

JOBID ===>

FOR JOB TO BE REQUEUED :

NEW OUTPUT CLASS ===> M

FOR JOB TO BE PRINTED : (A FOR ANSI)

PRINTER CARRIAGE CONTROL ===> A (M FOR MACHINE)

(BLANK FOR NONE)

شكل رقم (١٢-٦)

* ملاحظة :

١ ــ يمكن في المرات التالية ان نكتب مباشرة الخيار 5.2 فتظهر الشاشة ١٢ــ٢ ثم نكتب اسم البرنامج ونضغط ENTER و بالتالي نأخذ اسم العمل JOB-ID دون الحاجة لكتابة عبارات JCL مرّة ثانية.

٢ ــ قد يحصل في المرات الأولى أن يظهر JCL ERROR أخطاء في لغة التحكم، فيرجع للخيار 5 لتصحيح العبارات.

ربط البرنامج LINK باستخدام الخيار 5.7 :

بعد الانتهاء من تنقيح الأخطاء القواعدية من البرنامج، يحتاج هذا البرنامج للتنفيذ و يتم ذلك بعمل LINK لمجموعة البرامج الجزئية ووضع الناتج بملف DATASET تدعى LOAD، ولكل برنامج له صيغة LOAD صيغة التحميل من أجل التنفيذ، و يتم ذلك بكتابة الخيار 5.7 فتظهر الشاشة رقم (١٢ــ٥) نكتب اسم البرنامج مقابل للرمز شكل (١٢ــ٥) MEMBER من أجل ربط هذا البرنامج وتهيئته للتنفيذ.

وكما مرّ معنا سابقاً تعطي هذه الخطوة اسم JOB NAME، يرجع للخيار 3.8 من أجل استعراض ناتج عملية الربط، فلو كانت المحصلة أو رمز الشرط RETURN CODE هو الصفر ــ وهو مسجل في بداية العمل ــ فهذا يعني أنه تم ربط البرنامج بنجاح، وإلاّ نراجع JCL الخاص بالخيار رقم 5 أو نراجع البرنامج من أجل التأكد من خلوه من الأخطاء.

ويمكن الرجوع إلى المرجع MANUAL الخاص بالأخطاء لايجاد مصدر الخطأ.

تنفيذ البرنامج EXECUTE

بعد الانتهاء من ربط البرنامج بنجاح، يصبح البرنامج جاهز لعملية التنفيذ، وتتم عملية التنفيذ بكتابة عدة أوامر من لغة التحكم JCL كما هو واضح بالشكل التالي :

```
//STDXXXX     JOB    (STD, XXXX), 'NNNN', CLASS = H, MSGLEVEL = (1, 1),
//                   MSGCLASS = Y, NOTIFY = STDXXXX
// STP1               EXEC  PGM = programe name
//STEPLIB             DD    DSNAME = dataset, LOAD, DISP = SHR
// Input file name    DD    DSN = dataset name, DISP = SHR
// Out put file name  DD    DSN = dataset name, DISP = SHR
//
```

تدخل هذه الأسطر في الخيار 2 تحت اسم معين و يفضل أن يحتوي على الأحرف JCL ورمز عن اسم البرنامج الذي سينفذه لسهولة الرجوع إليه.

مع مراعاة ما يلي :

١ ــ الأسطر تبدأ بالموقع ١ ولا تتبع قواعد لغة الكوبول، بل تتبع قواعد لغة التحكم JCL.

٢ ــ تبدل XXXX بالرقم التعريفي، و NNNN باسم المبرمج.

٣ ــ إن عبارة Program name تدل على اسم البرنامج الذي سينفذ والذي تمت عملية ربطه LINKED في الخطوة السابقة في الخيار 5.7

٤ ــ يذكر في عنوان الفقرة STEPLIB ، اسم الملف الحاوي على الملف LOAD فغالباً ما يكون

DSNAME = USERID. TEST. LOAD

٥ ــ إن كلاً من Input file name و Output file name هما أسماء الملفات المذكورين في عبارة ASSIGN من SELECT ، فلو كان عندي المثال التالي :

SELECT INPUT F ASSIGN TO JFS - CARDF

فان الاسم CARDF هو الذي يجب أن يذكر في JCL أي

```
//CARDF  DD  DSN = USERID - TEST. CARDF, DISP = SHR...
```

أو أي اسم يعطى من قبل المشرف

نفس الكلام ينطبق على OUTPUT - FILE

وتجدر الاشارة أنه من المحتمل أن يكون في البرنامج أكثر من ملف ادخال وأكثر من ملف اخراج. فيجب أن يكون له عبارة DD (DATA DEFINITION) خاصة.

كما ويمكن الاّ يحتوي البرنامج على ملف ادخال، أو لا يحتوي على ملف اخراج، لكن المهم أنه لكل ملف يجب أن يكون له عبارة DD خاصة به.

أما عن اسم الملف الموجود على الوسط الثانوي فيعطى من قبل المشرف أو المسؤول عن تنظيم الملفات.

٦ ـــ في حال كون ملف الاخراج على الطابعة يكون شكل عبارة DD كالتالي :

```
// Out put file name     DD     SYSOUT = X
```

يوضع حرف يدل على نوع الطابعة المستخدمة و يرجع للمشرف

٧ ـــ قد تدخل البيانات مباشرة كبطاقات بعد بطاقة تعريف الملف و يكون شكلها كالتالي :

```
// CARDF DD    *
----------
----------
----------
```

بطاقات البيانات DATA

وتدل النجمة (*) على أن السجلات RECORDS التابعة للملف CARDF تلي هذا السطر مباشرة

٨ ــ من أجل زيادة المعلومات عن لغة التحكم بالعمل JCL يرجع للمرجع

OS/VS2 MVS JCL رقم GC28/0692/4

٩ ــ بعد الانتهاء من وضع كل المعايير المطلوبة نكتب الأمر SUBMIT في السطر الثاني من الشاشة مقابل عبارة<INPUT COMMAND نضغط ENTER فيرسل العمل للتنفيذ و يعطى اسم عمل JOBNAME معين وغالباً ما يكون الاسم والرقم التعريفي USERID مضافاً إليه الحرف S ــ يمكن ان يكون غير S ــ ، ونخرج من برنامج EDIT بواسطة مفتاح PF15 عدة مرات.

الكشف عن ناتج التنفيذ :

بعد الانتهاء من التنفيذ، نذهب للخيار 3.8 للكشف عن ناتج العمل نكتب اسم العمل الذي أخذناه من الخطوة السابقة، ونستعرضه فاذا لم يحتوي على أخطاء قواعدية في JCL فهذا يعني أن البرنامج قد نفذ بنجاح، فاذا كان الناتج على الطابعة مثلاً نكتب الخيار P أو L من أجل تنفيذ الطباعة، أما إذا كان الناتج على قرص ممغنط، فيمكن استعراض الناتج في خيار آخر هو الخيار (1)BROWSE ــ اعتباراً من الشاشة الأساسية رقم ١١ ــ ١.

إذا كان ناتج البرنامج OUTPUT يحوي على أخطاء منطقية أي ليست النتيجة المرجوه فهذا يعني خطأ في البرنامج، فيرجع للرسم الانسيابي FLOWCHART و يدرس و يعدّل إذا لزم الأمر، ثم إلى صفحات الترميز CODING فتعدّل ، ثم يعدّل البرنامج الموجود في الحاسب باستخدام منقح النصوص TEXT EDITOR ، و يعمل له ترجمة ثم ربط ثم تنفيذ و يدُرس الناتج من جديد. فاذا كان خالياً من الأخطاء المنطقية فهذا يعني أن البرنامج قد حقق المطلوب منه وإلاّ تعاد الخطوات السابقة من جديد.

* توثيق البرنامج :

بعد الانتهاء من تنفيذ البرنامج والتأكد من النتائج ومناقشتها مع محلل النظم أو المشرف وإقرارها، يطبع نسخه من البرنامج — أو عدّة نسخ — وذلك بالخيار 3.6 أو 3.8، وتحفظ نسخة من البرنامج مع المخطط الانسيابي له FLOWCHART مع أوراق الترميز CODING SHEET بالاضافة لنماذج ادخال أو نماذج اخراج أو نتائج في ملف بلاستيكي و يعطى له عنوان دال على محتواه.

و يفضل أن يحتوي البرنامج على ملاحظات COMMENTS تشير إلى خطوات العمل كما و يفضل كتابة الخطوات التي اتبعت لحل هذا البرنامج وتحفظ في ملف البرنامج و يُسلم إلى محلل النظم المسؤول أو مدير المشروع أو المشرف عند الطلب.

١٢ - ٣ ترجمة وتنفيذ البرنامج باستخدام الحاسب TI990 نظام التشغيل DX10

يستدعى المترجم COMPILER في الحاسب TI990 بكتابة الأمر التالي :

X CCF =

EXECUTE COBOL COMPILER FOREGROUND

فتظهر لدينا الشاشة التالية :

SOURCE ACCESS NAME : اسم البرنامج الذي نرى

OBJECT ACCESS NAME : عمل له COMPILE

LIST ACCESS NAME :

OPTIONS : RETURN اضغط

PRINT WIDTH : 80 RETURN اضغط

---

* قد تحتوي بعض مراكز الكمبيوتر على نماذج توثيق خاصة، فتستعمل عندئذٍ حسب توجيهات كل مركز.

اضغط RETURN حجم الصفحة PAGE SIZE : 55

اضغط PROGRAM SIZE (LINE) : 1000 RETURN

يكتب في السطر الأول اسم البرنامج المراد: ترجمته وفي حالتنا :

اسم البرنامج .STDOOL, COBOL 3, SRC *

وفي السطر الثاني اسم البرنامج بشكل التنفيذ وفي حالتنا هنا :

اسم البرنامج .STDOOL. COBOL 3, OBJ

وفي السطر الثالث اسم الملف الذي سيوضح فيه قائمة بالبرنامج و بالعناوين المستخدمة و بالأخطاء إن وجدت والتي سيرجع إليها المبرمج لتحري الأخطاء وفي حالتنا هذه يكون :

اسم البرنامج .STDOOL. COBOL 3. LST

أما بقية الأسطر فهي عرض سطر الطباعة. وعدد الأسطر في الصفحة، وحجم البرنامج بالأسطر فكلها جاهزة فنضغط RETURN فقط.

ملاحظة هامة :

إن صيغة الأصل SOURCE هي التي تعدل فقط

أما صيغة التنفيذ OBJECT فليس للمبرمج عمل عليها.

أخيراً صيغة الـ LIST هي فقط لاستعراض الأخطاء ولا يعدّل عليها.

تنفيذ البرنامج على الحاسب TI990

يتم تنفيذ البرنامج بعد عمل COMPILATION له وتشكيل OBJECT ، إن صيغة التنفيذ OBJECT هي وحدها التي تنفذ فقط دون صيغة الأصل أو الـ LIST

* أسماء الملفات والوحدات يمكن أن تتبدل من موقع لآخر، فيرجع للأستاذ المشرف من أجل أخذ الأسماء المطبقة.

فبعد التأكد من خلو البرنامج من الأخطاء القواعدية يشرع في تنفيذ البرنامج علينا تجهيز الملفات FILES وتتم بالشكل التالي :

وضمن برنامج الكوبول هناك لكل ملف في قسم بيئة البرنامج ENVIRONMENT DIVISION هناك تعليمة اختيار الملفات وهي :

SELECT file name ASSIGN TO { INPUT / OUTPUT / RANDOM / PRINT } , "SYN"

إن ما بين الفاصلتين العلويتين هناك الأسم المختصر و يدعي SYNONAM هذا الأسم هو صلة الوصل بين البرنامج والوسط الموجود عليه الملف هل هو موجود على DISK أو TAPE أو PRINTER أو CARD READER

فقبل الشروع بتنفيذ البرنامج نعين الأسم المختصر بتعليمة

ASSIGN SYNONYM **AS**

فتظهر لنا الشاشة التالية :

ASSIGN SYNONYM

SYNONYM : الأسم المختصر الموجود في البرنامج ضمن الفاصلتين العلويتين

VALUE : قيمة هذا الأسم أي على أي وسط موجود ؟

مثال : لو كان عندي ملف طباعة يكون توظيفه في ملف البرنامج كالتالي :

SELECT PRFL ASSING TO PRINT, "PRT". برنامج الكوبول

فالرمز PRT هو صلة الوصل بين البرنامج والطابعة فنكتب الاسم المختصر كالتالي : ASSIGN SYNONYM

SYNONYM : PRT الاسم الموجود بالبرنامج

VALUE : ST01 اسم الطابعة

إذاً فالاسم PRT يعادل أو يدل على الطابعة التي اسمها ST01 مثلاً .

مثال : لو كان عندي ملف موجود على القرص الممغنط DISC أكتب :

في برنامج الكوبول SELECT DSCFL ASSIGN TO INPUT, "DFL"

ونشير إليه بالاسم المختصر كالتالي :

ASSIGN SYNONAM

الاسم المختصر الموجود في البرنامج SYNONAM : DFL

الاسم الموجود على الديسك VALUE : STD001. COBOL 3. FILES. TOFL

بعد تعيين الـ SYNONAM نكتب التعليمة التالية لتنفيذ البرنامج :

XCPF

EXECUTE COBOL PROGRAM FOREGROUND

فتظهر لنا الشاشة التالية :

OBJECT ACCESS NAME :

اضغط DEBUG MODE : NO RETURN

اضغط MESSAGE ACCESS NAME : RETURN

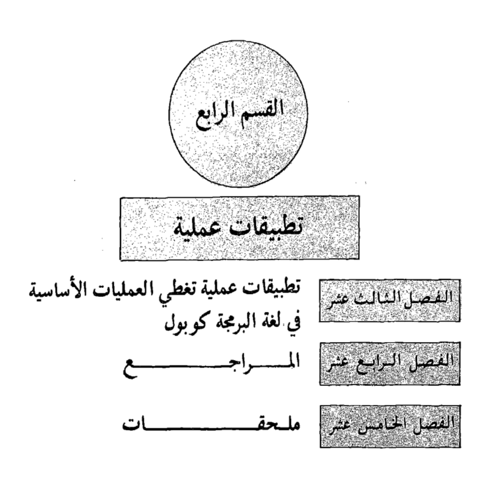
اضغط SWITCHS : 00000 RETURN

اضغط FUNCTION KEYS : NO RETURN

نكتب اسم البرنامج في السطر الأول ، و يشترط أن يكون اسم البرنامج بصيغة التنفيذ أي في حالتنا هنا : اسم البرنامج STD001. COBOL 3. OBJ.

ونضغط RETURN لباقي الأسطر

ننتظر قليلاً لبدء تنفيذ البرنامج ، في حال ظهور رسائل تدل على الخطأ استدع الأستاذ المشرف ، أما في حال حصول على نتائج خاطئة نرجع للبرنامج ونعدّله بعد دراسة الخطأ ونعاود ترجمته COMPILE ثم تنفيذه وهكذا ....


القسم الرابع
تطبيقات عملية
الفصل الثالث عشر
تطبيقات عملية تغطي العمليات الأساسية في لغة البرمجة كوبول
الفصل الرابع عشر
المـــــراجـــــــع
الفصل الخامس عشر
ملـحقـــــــــات

الفصل الثالث عشر

تطبيقات عملية للعمليات الأساسية في لغة البرمجة كوبول

١٣ ــ ١ لمحة عامة :

يحتوي هذا الفصل على التطبيقات الأساسية التي تفيد المبرمج أو الدارس في وضع وتصميم البرامج في المستقبل أو يمكن لهذه التطبيقات أن تعطي صورة أساسية لمختلف العمليات التي تنفذ على الملفات والبيانات باستخدام كلا النوعين من الحاسبات الصغيرة والكبيرة منها أي باستخدام الشاشات المتلفزة والتعليمات المتطورة الخاصة بها وباتباع أسلوبي التشغيل المعروفين: المباشر وغير المباشر حتى يتمكن المبرمج من الانطلاق في تطوير وتصميم برامج أكثر تعقيداً وأشمل معالجة، وليست هذه البرامج سوى نماذج عملية لتنير له الطريق للتقدم والتطور وفهم هذه اللغة واستخدامها من الناحية العملية بصورة أفضل.

١٣ ــ ٢ برنامج طباعة تقرير:

التمرين التالي وهو طباعة ملف متسلسل SEQUENTIAL مفروز على رقم المشترك D-No

إن عمليات الطباعة من العمليات السهلة نسبياً، والعنصر الأساسي فيها هو توصيف أسطر الطباعة وكما نعلم يشمل على ٣ أجزاء :

أ ــ العنوان PAGE HEADING

ب ــ جسم التقرير PAGE BODY

جـ ـــ أسفل الصفحة PAGE FOOTING

أما بالنسبة للعناوين فتوصف في فصل التخزين الداخلي للبرنامج WORKING STORAGE SECTION بمعدل سطر توصيف لكل سطر موجود على ملف شكل الطباعة المعطى سلفاً PRINT LAYOUT SHART و يلاحظ أن يوصف مكان معين للتاريخ (سنة ـــ شهر ـــ يوم) وكذلك لعداد رقم الصفحات.

وتعطى المتغيرات التي ستدخل مع السطر أسماء متغيرات أو حقول بينما توصيف باقي السطر فيوصف كـ FILLER

أما جسم التقرير فهو عبارة عن سطر غالباً يكرر على طول الصفحة يحتوي على حقول متغيرة كل مرّة فيوصف سطر واحد هو سطر عمل وتحرك له البيانات ثم يطبع.

أما أسفل الصفحة فقد ينص البرنامج صراحة على طباعة مجاميع في أسفل الصفحة عندئذ يوصف سطراً مستقلاً له. وإذا لم يستعمل فيمكن لجسم الصفحة أن يستغل أسفل الصفحة.

ومن أجل قراءة المخرجات بصورة مريحة وصحيحة يترك فراغ واحد على الأقل بين سطري طباعة، و يراعى أن تكون المسافات الافقية بين الحقول كافية.

وغالباً ما تعالج طباعة العنوان في عنوان فقرة مستقلة تتكرر عندما تكون قيمة عداد الأسطر مساوية لقيمة معينة ينص عليها البرنامج صراحة.

ومن الأخطاء الشائعة في برامج الطباعة أن المبرمج ينسى أن يحرك القيمة صفر إلى عداد الأسطر و بالتالي يتكرر عنده العنوان أكثر من مرّة بشكل خاطيء، لذا يجب تحريك القيمة صفر للعداد كلما قلبنا صفحة.

١٣ - ٢ - ١ نص التمرين :

اكتب برنامجاً بلغة كوبول يقرأ ملف المشتركين SUBFILE و يطبع محتواه على الطابعة علماً بأن الملف موجود على قرص ممغنط.

مع مراعاة ما يلي :

١ ــ اطبع العناو ين والتاريخ ورقم الصفحة في رأس كل صفحة.

٢ ــ لا تطبع في الصفحة الواحدة أكثر من ٢٥ سطر ــ عدا العناو ين.

٣ ــ اترك سطراً فارغاً بين كل سطري طباعة.

* في الصفحة التالية يوجد توصيف الطابعة :

توصيفة	اسم الحقل
9 (5)	D-NO
X (2ϕ)	D-NAME
X (24)	D-ADDRS
9 (8)	D-DATE
99	D-CODE
99	D-AREA
9 (7)	D-CNTER
9 (7)	D-LREAD
X (5)	D-FILLER

أنظر الصفحة رقم ٦٠٧ لتوصيف شكل المخرجات والصفحة ٥٩٧ للنتائج.

```
*        حل تمرين طباعة تقرير ١٣-٣-١
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT06.
 AUTHOR.       KHAYAT-JNAID.
*
*  يرى هذا التمرين الخطوات الاساسيه في كتابه
*  برنامج يطبع تقرير :
*
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.    IBM-370.
   OBJECT-COMPUTER.    IBM-370.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
       SELECT SUBFILE ASSIGN  TO UT-S-SFILE.
       SELECT P-FILE  ASSIGN  TO UT-S-PRFL.
 DATA DIVISION.
 FILE  SECTION.
*   توصيف ملف الادخال :
*
 FD     SUBFILE   LABEL RECORD  IS STANDARD
        BLOCK CONTAINS 0 RECORDS DATA RECORD IS SUB-REC
 01     SUB-REC.
        03      D-NO       PIC  9(5).
        03      D-NAME     PIC  X(20).
        03      D-ADDRS    PIC  X(24).
        03      D-DATE     PIC  X(8).
        03      D-CODE     PIC  99.
        03      D-AREA     PIC  99.
        03      D-CNTR     PIC  9(7).
        03      D-LREAD    PIC  9(7).
        03      FILLER     PIC  X(5).
*
*   توصيف ملف الطابعه :
*
 FD     P-FILE     LABEL RECORD  IS OMITTED
                        DATA RECORD IS  P-REC.
*
*يوصف غالبا سجل بطول ١٣٢ حرفا وهو عرض الطابعه
*WORKING-STORAGE وتحرك السجلات الاخرى الموصوفه في
*اليه ويتم الكتابه باسم هذا السجل P-REC ومايحتوى
*  من بيانات مختلفه .
*
```

```
 01     P_REC          PIC     X(132).
*
 WORKING_STORAGE SECTION.

*-----------------------------------------------------------*
*   توصف العناوين كما هو واضح في ورقة                       *
*   التوصيف المرفقة PRINT CHART وتسمى                       *
*   العناوين الثابتة او الفراغات                             *
*   ب FILLER بينما الحقول المتغيرة                          *
*   مثل التاريخ ورقم الصفحة باسماء                          *
*   مناسبة تدل على محتوى الحقل لتسهيل                       *
*   معالجته___________________________________              *
*-----------------------------------------------------------*

 01     HDR_1.
  03    FILLER    PIC  X(58) VALUE SPACES.
  03    FILLER    PIC  X(16) VALUE
   'ELECTRIC COMPANY'.
  03    FILLER    PIC  X(58) VALUE SPACES.
*
 01     HDR_2.
  03    FILLER    PIC  X(08) VALUE 'DATE:'.
  03    DAY_P     PIC  99.
  03    FILLER    PIC  X     VALUE '/'.
  03    MONTH_P   PIC  99.
  03    FILLER    PIC  XXX   VALUE '/19'.
  03    YEAR_P    PIC  99.
  03    FILLER    PIC  X(35) VALUE SPACES.
  03    FILLER    PIC  X(26) VALUE
       'LIST OF SUBSCRIBER RECORDS'.
  03    FILLER    PIC  X(39) VALUE SPACES.
  03    FILLER    PIC  X(10) VALUE
        ' PAGE NO.:'.
  03    P_CTR     PIC  ZZ9.
  03    FILLER    PIC  X(10) VALUE SPACES.
*
 01     HDR_3.
  03    FILLER    PIC  X(17) VALUE
        '  SUBSCRIBER NO.'.
  03    FILLER    PIC  X(13) VALUE
        '    SUBS. NAME'.
```

```
       03    FILLER    PIC  X(20) VALUE SPACES.
       03    FILLER    PIC  X(13) VALUE
             'SUBS. ADDRESS'.
       03    FILLER    PIC  X(11) VALUE SPACES.
       03    FILLER    PIC  X(17) VALUE
             'SUBSCRIPTION DATE'.
       03    FILLER    PIC  X(13) VALUE
             '  SUBS. CODE'.
       03    FILLER    PIC  X(14) VALUE
             '    SUBS. AREA '.
       03    FILLER    PIC  X(14) VALUE
             'COUNTR LAST RD'.

*-----------------------------------------------------------------*
* السطر التالي هو لتوصيف سطر جسم التقرير                          *
* اي السطر الذي يحتوى على BODY LINE                               *
* البيانات المتغيره والذى سيتكرر على طول                          *
* الصفحة                                                          *
*-----------------------------------------------------------------*

      01     DATA-LINE.
       03    FILLER    PIC  XX     VALUE SPACES.
       03    S-NO      PIC  X(5).
       03    FILLER    PIC  X(08) VALUE SPACES.
       03    S-NAME    PIC  X(20).
       03    FILLER    PIC  X(10) VALUE SPACES.
       03    S-ADDRS   PIC  X(24).
       03    FILLER    PIC  X(10) VALUE SPACES.
       03    S-DATE    PIC  99/99/9999.
       03    FILLER    PIC  X(08) VALUE SPACES.
       03    S-CODE    PIC  X(2).
       03    FILLER    PIC  X(12) VALUE SPACES.
       03    S-AREA    PIC  X(2).
       03    FILLER    PIC  X(4)  VALUE SPACES.
       03    S-CNTR    PIC  Z(5)99.
       03    FILLER    PIC  X(02) VALUE SPACES.
       03    S-LREAD   PIC  Z(5)99.
       03    FILLER    PIC  X      VALUE SPACES.
*
*  لقبول التاريخ من الحاسب
```

```
 01     CURENT-DATE.
  03    C-YEAR   PIC     99.
  03    C-MONTH  PIC     99.
  03    C-DAY    PIC     99.
*
*     عدادات تستخدم لتجميع
*  عدد الاسطر وعدد الصفحات
 77     LN-CTR   PIC     99        VALUE 0.
 77     PG-CTR   PIC     999       VALUE 0.
*
 PROCEDURE DIVISION.
*
*  الخطوه الاولى هى فتح الملفات
        OPEN      INPUT   SUBFILE
                   OUTPUT  P-FILE.
*
*قبول التاريخ من الحاسب وتحريكه
*        الى حقول الطباعه ثم طباعه
*   العناوين فى الفقره NEW-PAGE,
*
        ACCEPT    CURENT-DATE FROM DATE.
        MOVE      C-YEAR         TO    YEAR-P.
        MOVE      C-MONTH        TO    MONTH-P.
        MOVE      C-DAY          TO    DAY-P.
        PERFORM   NEW-PAGE.
*
 MAIN-LOOP.
        READ    SUBFILE   AT END
                  CLOSE SUBFILE P-FILE STOP RUN.
*
*
*       تحريك الحقول من ملف الادخال SUBFILE
*  DATA-LINE المقروء الى حقول سجل الطباعه
*    الموصف فى قسم توصيف البيانات الداخليه
        MOVE      D-NO        TO        S-NO.
        MOVE      D-NAME      TO        S-NAME.
        MOVE      D-ADDRS     TO        S-ADDRS.
        MOVE      D-DATE      TO        S-DATE.
        MOVE      D-CODE      TO        S-CODE.
        MOVE      D-AREA      TO        S-AREA.
        MOVE      D-CNTR      TO        S-CNTR.
        MOVE      D-LREAD     TO        S-LREAD.
```

```
*
*
*   :  كتابة السجل على ملف الطباعة
*
       WRITE    P_REC     FROM    DATA_LINE
                          AFTER ADVANCING 2 LINES.
*
*
*   اضافة ١ لعداد الاسطر LN_C ثم مقارنة
*      محتواه مع العدد ٢٥ لان عدد الاسطر
*   المسموح طباعتها في الصفحة الواحدة
*   هو ٢٤ , فاذا كان اكبر تقلب الصفحة
*   وتطبع العناوين على الصفحة الجديدة
*   ثم يذهب التنفيذ الى قراءة سجل جديد ,
*   واذا لم يكن اكبر فيذهب التنفيذ الى
*   معالجة سجل جديد مباشرة .
*
       ADD      1          TO       LN_CTR.
       IF       LN_CTR     >        25
                           PERFORM   NEW_PAGE.
       GO TO    MAIN_LOOP.
*
*
*   فقرة قلب الصفحة وكتابة العناوين :
*   ( لاحظ اضافة ١ لعداد الصفحات C_CNTR
*   وتحريك القيمة صفر لعداد الاسطر )
 NEW_PAGE.
       MOVE     ZERO       TO       LN_CTR.
       ADD      1          TO       PG_CTR.
       MOVE     PG_CTR     TO       P_CTR.
       MOVE     SPACES     TO       P_REC.
       WRITE    P_REC      AFTER    PAGE.
       WRITE    P_REC FROM HDR_1 AFTER 1 LINE.
       WRITE    P_REC FROM HDR_2 AFTER 1 LINE.
       WRITE    P_REC FROM HDR_3 AFTER 2 LINE.
  EXTP.
       EXIT.
```

١٣ ــ ٣ برنامج معالجة بيانات عن طريق الأسلوب المباشر :

البرنامج التالي يبين انشاء ملف مفهرس عن طريق استخدام الشاشات TERMINALS وتستخدم غالباً في الأجهزة الصغيرة مثل جهاز تكساس انسترومنتس، بينما الأجهزة الكبيرة مثل IBM فان العمل عليها يمر غالباً باستخدم طرق المعالجة غير المباشرة BATCH MODE لأن استخدام الشاشات TERMINALS يحدد باستخدام برامج ونظم خاصة مثل نظام 'CICS' في جهاز IBM وهو بعيد قليلاً عن لغة كوبول.

والمثال التالي يوضح كيفية كتابة برنامج ينشيء ملف مفهرس باستخدام الشاشات. إذ توصف الشاشات ضمن برنامج كوبول باستخدام تعليمة DISPLAY ومعاييرها المختلفة لاظهار الرسائل في السطر المعين والموقع المحدد الذي يكون موجوداً أصلاً على ورقة توصيف خاصة بالشاشات.

يحدد المبرمج الشاشات المختلفة التي يريدها. فيبني شاشة الاختيارات الأولى وتسمى عادة MAIN MENU وفيها العمليات الأساسية التي ستجري في البرنامج وغالباً ما تكون: إضافة سجل، تعديل سجل، حذف سجل، استفسار...

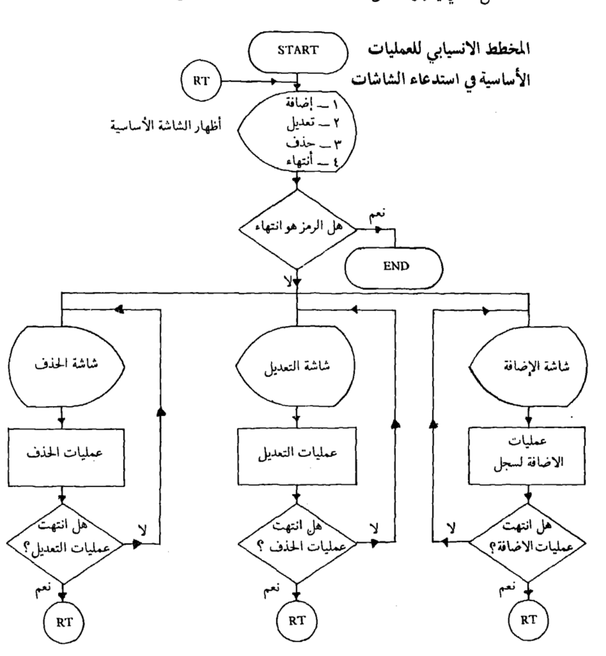
ومن هذه الشاشة تدخل رمز العملية المطلوبة لينتقل التنفيذ منها إلى الشاشات أخرى خاصة بهذا الخيار.

وغالباً ما تكون الشاشات موضح فيها كل العمليات الأساسية التي سيقوم بها مشغل النظام ومن النصائح الأساسية التي يجب مراعاتها عند تصميم الشاشات أن تظهر الرسائل المناسبة عند الضرورة والاّ يترك المشغل حائراً ماذا سيفعل ؟ إذ يجب أن يظهر المبرمج رسائل محددة تدل المشغل على الخطوة التالية. مثل (السجل غير موجود، حاول ثانية) الخ... من الرسائل التي تغني جزئياً عن دليل التشغيل USER GUIDE

ويمكن أن يطلب عمل تحقق للحقول الداخلة عندها عند قبول كل حقل يعمل تحقق عليه وتظهر الرسائل المناسبة في حال الخطأ أمام المدخل بصورة خاطئة.

و يـنهى البرنامج غالباً برمز معين يدخل ضمن الشاشة الرئيسية وليس هناك شرط لنهاية ملفات أو غيره غالباً.

الشكل التالي يظهر تسلسل استدعاء الشاشات ضمن البرنامج.



١٣ - ٣ - ١ نص التمرين :

المطلوب كتابة برنامج بلغة الكوبول ينشيء (CREATE) الملف الأساسي للمشتركين واسمه SUBSFL على أن يكون هذا الملف من نوع المفهرس INDEX SEQUENTIAL وذلك عن طريق الشاشات TERMINALS (وفق النموذج المرفق ــ حالة TI).

كما و يطلب أن يكون البرنامج يسمح بعمليات التحديث UPDATE على الملف، والتي تشمل تعديل سجل CHANGE أو إضافة INSERT أو حذف سجل DELETE

وتتم العمليات وفق ما يلي :

أ ــ مرحلة إضافة سجلات جديدة INSERT NEW RECORDS :

يقبل عن طريق الشاشة TERMINAL المعلومات التالية :

١ ــ رقم المشترك SUBS-NO ويجب أن يكون رقمياً وأكبر من الصفر.

٢ ــ اسم المشترك SUBS-NAME يجب الاّ يكون رقمياً أو فراغ.

٣ ــ عنوان المشترك SUBS-ADRES يجب الاّ يكون فارغ.

٤ ــ تاريخ الأشتراك SUBS-DATE :

SUBS-DAY اليوم يجب أن يكون محصوراً بين ١ ــ ٣٠.

SUB-MONTH الشهر يجب أن يكون محصوراً بين ١ ــ ١٢.

SUBS-YEAR السنة يجب أن تكون بين ١٤٠١هـ ــ ١٤١٠هـ.

٥ ــ رمز الأشتراك SUBS-CODE يجب أن يكون رقمياً ومحصوراً بين ١ ــ ١٠.

٦ ــ رمز المنطقة SUBS-AREA يجب أن يكون رقمياً ومحصوراً بين ١ ــ ٢٠.

٧ ــ رقم العداد SUBS-CNTR يجب أن يكون رقمياً وأكبر من الصفر.

٨ ــ مقدار القراءة SUBS-LREAD يجب أن يكون رقمياً وأكبر من الصفر.

عند قبول الحقول السابقة يجب عمل تحقق VALIDATION حسب القواعد المذكورة سابقاً، وعند وجود أي حقل خطأ أظهر عبارة 'ERROR' أمام الحقل الخطأ، وكرر قبول الحقل من جديد. وعند تكرار الخطأ ثلاثة مرات للحقل الواحد أرفض السجل وأظهر رسالة 'REJECTED RECORD' واذهب لتقبل سجل جديد.

أما عند خلو الحقول من الأخطاء أكتب هذا السجل على الملف SUBSFL وفي حال نجاح الكتابة أظهر رسالة 'SUCCESSFUL INSERTION' وكرر العمل من أجل سجل جديد، أما في حالة الخطأ INVALID KEY CONDITION فأظهر رسالة 'DUPLICATE KEY' وكرر العمل من أجل سجل جديد.

في نهاية الاضافة ادخل الرمز المناسب للخروج من حالة الاضافة والذهاب إلى الشاشة الرئيسية MAIN MENU في البرنامج.

ب ـ تعديل سجل UPDATE :

و يطلب في هذه المرحلة قبول رقم المشترك المراد التعديل على بياناته.

ثم يقرأ سجل المشترك وفق الرقم المعطى عن طريق الشاشة.

في حال كون القراءة خاطئة أظهر عبارة 'RECORD NOT EXIST' وأذهب لتقبل رقم جديد.

أما في حال كون القراءة ناجحة فأظهر محتوى هذا السجل المقروء على الشاشة وفق النموذج المرفق وأظهر في أسفل الشاشة عبارة :

'ENTER THE FIELD NUMBER :'

أي ادخل رقم الحقل المراد التعديل عليه : أقبل رقم الحقل وعدّل الحقل المرافق.

ولا تنسى عمل تحقق VALIDATION له. وفي النهاية أدخل الرمز المناسب للانتهاء من تعديل هذا السجل أو أعد كتابة السجل على الملف.

كـرر الـعمل من أجل سجل آخر، وفي نهاية عمليات التعديل أدخل الرمز المناسب للعودة للشاشة الرئيسية في البرنامج.

جـ ــ عملية حذف سجل DELETE A RECORD :

١ ــ اقبل عن طريق الشاشة رقم المشترك المراد حذف سجله.

٢ ــ اقرأ السجل بالمفتاح السابق.

وفي حال القراءة خاطئة أظهر رسالة 'RECORD NOT EXIST' السجل غير موجود وأذهب للخطوة رقم ١.

أما اذا كان السجل موجود فأظهره على الشاشة وأظهر رسالة:

'ARE YOU SHURE ? Y/N' هل أنت متأكد؟ ن/لا

فـاذا كـانـت الاجـابـة لا أي أنه ليس هذا السجل المطلوب فارجع للخطوة رقم ١. والاّ احذف هذا السجل وأذهب للخطوة رقم ١.

٣ ــ في نـهـاية عمليات الحذف ادخل الرمز المناسب للخروج من عمليات الحذف والرجوع للشاشة الرئيسية MAIN MENUE

ادخل الرمز المناسب في الشاشة الرئيسية للخروج من البرنامج.

توصيف الشاشة الأساسية :

SELECT ONE OPTION :

1. INSERT NEW RECORD.
2. UPDATE EXIST RECORD.
3. DELETE EXIST RECORD.
4. EXIT

FILE NAME PIC : SUBSFL توصيف الملف

رقم المشترك	SUBS-NO	X (5)	
اسم المشترك	SUBS-NAME	X (30)	
عنوانه	SUBS-ADRESS	X (24)	
تاريخ الأشتراك	SUBS-DATE		
يوم	SUBS-DAY		99
شهر	SUBS-MONTH		99
سنة	SUBS-YEAR		9 (4)
رمز الأشتراك	SUBS-CODE	99	
رمز المنطقة	SUBS-AREA	99	
رقم العداد	SUBS-CNTR	9 (7)	
مقدار آخر قراءة	SUBS-LREAD	9 (7)	
فــراغ	SUBS-FLR	X (4)	

١٣ ـ ٣ ـ ٢

نص برنامج معالجة بيانات :

```
IDENTIFICATION DIVISION.
PROGRAM-ID.     ONLNUPD.
AUTHOR.         KHAYAT-JNAID.
*
ENVIRONMENT     DIVISION.
CONFIGURATION   SECTION.
SOURCE-COMPUTER. TI990.
OBJECT-COMPUTER. TI990.
INPUT-OUTPUT    SECTION.
FILE-CONTROL.
        SELECT SUBSFL ASSIGN TO RANDOM "SM"
        ORGANIZATION  IS     INDEXED;
        ACCESS MODE   IS     RANDOM;
        RECORD KEY    IS     SUBS-NO.
*
DATA           DIVISION.
FILE           SECTION.
```

```
 FD    SUBSFL    LABEL RECORD STANDARD
                 RECORD CONTAINS 80 CHARACTERS.
 01      SUBS-REC.
     03 SUBS-NO            PIC    X(05).
     03 SUBS-NAME          PIC    X(20).
     03 SUBS-ADRES         PIC    X(24).
     03 SUBS-DATE.
        05   SUBS-DAY      PIC 99.
        88   DAYY       VALUES ARE 01 THRU 30.
        05   SUBS-MONTH PIC 99.
        88   MONTHH   VALUES ARE 01 THRU 12.
        05   SUBS-YEAR   PIC 9999.
        88   YEARR      VALUES ARE 1401 THRU 1410.
     03 SUBS-CODE          PIC 99.
     88  CODEE          VALUES ARE 01 THRU 10.
     03 SUBS-AREA          PIC 99.
     88  AREAA          VALUES ARE 01 THRU 20
     03 SUBS-CNTR          PIC 9(7).
     03 SUBS-LREAD         PIC 9(7).
     03 FILLER             PIC X(5).
*
 WORKING-STORAGE        SECTION.
*
 01    DEC-CODE            PIC 99
                 USAGE IS COMP-1.
 01    BIN-VAL REDEFINES DEC-CODE PIC XX.
 01    CHAR                PIC 99.
 77    FLAG                PIC 9 VALUE ZERO.
 77    FLD-NO              PIC 99.
 77    DUMMY               PIC XX.
 77    ANSR                PIC X.
 77    OPTIONS             PIC 9.
*
 PROCEDURE         DIVISION.
 STRT.
          OPEN    I-O SUBSFL.
* التعليمات الثلاث التاليه تستخدم لتحويل
* نمط الكتابه على الشاشه الى العربى
*
          MOVE  18       TO DEC-CODE.
          MOVE BIN-VAL  TO CHAR.
          DISPLAY           CHAR.
*
 MENU.
       DISPLAY

          "برنامج بناء وتحديث الملف الاساسى"
                    LINE 3 POSITION 24 ERASE
                    "الشركه الكهرباء"
                     LINE 4 POSITION 30
          "---------------------------------"
                     LINE 5 POSITION 24
                    "اضافه سجل جديد-1"
                     LINE 6 POSITION 27
                    "تعديل سجل موجود-2"
```

```
                    LINE 8 POSITION 27
                   "حذف سجل موجود-3"
                    LINE 10 POSITION 27
                   "الاستفسار عن سجل-4"
                    LINE 12 POSITION 27
      "ادخل رقم العمليه او اضغط RETURN للانتهاء"
                    LINE 14 POSITION 20
      "-----------------------------------------"

                    LINE 16 POSITION 20.
ACPT-COD.
         ACCEPT OPTIONS
             LINE 14 POSITION 70 PROMPT TAB.
         IF OPTIONS LESS    THAN 1 OR
            OPTIONS GREATER THAN 4

         CLOSE SUBSFL STOP RUN.


*
*                    قبول رمز العمليه من الشاشه ،
*  ومعرفه هل نحن امام اضافه ام حذف ام تعديل
*
     MOVE HIGH-VALUE TO SUBS-NO.
     IF OPTIONS = 1 PERFORM INSRT-RTN THRU EXT-INS
                        UNTIL SUBS-NO = "00000".
     IF OPTIONS = 2 PERFORM UPDAT-RTN THRU EXT-UPD
                        UNTIL SUBS-NO = "00000".
     IF OPTIONS = 3 PERFORM DELT-RTN  THRU EXT-DEL
                        UNTIL SUBS-NO = "00000".
     IF OPTIONS = 4 PERFORM INQR-RTN  THRU EXT-INQ
                        UNTIL SUBS-NO = "00000".
         GO TO MENU.



*
* الفقره الخاصه بالتعديل وفيها تتم قراءه السجل المطلوب
* ثم تظهر معلوماته على الشاشه وتعدل حقوله حسب المطلوب
* REWRITE          وتعاد كتابه السجل على الملف بتعليمه
*
UPDAT-RTN.
          PERFORM ACCEPT-NO THRU EXT-AC.
          IF SUBS-NO = "00000" GO TO EXT-UPD.
          IF FLAG     = 1
             DISPLAY       "الســـجل غير موجود"
                    LINE 20 POSITION 10
                    ACCEPT  DUMMY GO TO EXT-UPD.
          PERFORM   DISPLAY-SCREEN THRU EXT-SC.
          PERFORM   DISPLAY-FIELDS THRU EXT-FLD.
          DISPLAY
          "ادخل رقم الحقل المراد تعديله او الرقم صفر للانتهاء"
          LINE      23 POSITION 5.
          PERFORM   ACCEPT-FLD THRU EXT-UPD
                         UNTIL FLD-NO = ZERO.
          REWRITE   SUBS-REC.
          DISPLAY
          "الســــــــجل عـــــــــدل"
          LINE      23 POSITION 5.
```

```
          ACCEPT    DUMMY.
          GO TO     UPDAT-RTN.
 ACCEPT-FLD.
      ACCEPT   FLD-NO LINE 23 POSITION 75 PROMPT  TAB.
      IF FLD-NO = 2      PERFORM AC-NAME       ELSE
      IF FLD-NO = 3      PERFORM AC-ADRES      ELSE
      IF FLD-NO = 5      PERFORM AC-YEAR       ELSE
      IF FLD-NO = 6      PERFORM AC-MONTH      ELSE
      IF FLD-NO = 7      PERFORM AC-DAY        ELSE
      IF FLD-NO = 8      PERFORM AC-CODE       ELSE
      IF FLD-NO = 9      PERFORM AC-AREA       ELSE
      IF FLD-NO = 10     PERFORM AC-CNTR       ELSE
      IF FLD-NO = 11     PERFORM AC-LREAD.
 EXT-UPD.
          EXIT.

*
* الفقره التاليه فى لاضافه سجل جديد وعمل تحقق عليه
*        ثم الكتابه على السجل بتعليمه WRITE
*
 INSRT-RTN.
          PERFORM ACCEPT-NO THRU EXT-AC.
          IF SUBS-NO = "00000" GO TO EXT-INS.
          IF FLAG     = ZERO DISPLAY
              "السجل مكرر فى الملف الرجاء اعد العمل"
              LINE 20 POSITION 10
              ACCEPT DUMMY  GO TO EXT-INS.
          PERFORM DISPLAY-SCREEN THRU EXT-SC.
          DISPLAY SUBS-NO LINE 3 POSITION 45.
*
* فى الفقرات التاليه يتم قبول كل حقل ويعمل تحقق عليه وتظهر
*        رساله خطا فى حل حدوثه يعاد قبوله من جديد
* ### لاحظ استخدام اسم الشرط عند التحقق من حقول التاريخ مثلا
 AC-NAME.
        ACCEPT SUBS-NAME LINE 5 POSITION 45 PROMPT TAB.
        IF      SUBS-NAME = SPACES OR SUBS-NAME NUMERIC
                                            GO TO   AC-NAME.
*
 AC-ADRES.
        ACCEPT SUBS-ADRES LINE 6 POSITION 45 PROMPT TAB.
        IF SUBS-ADRES =    SPACES GO TO AC-ADRES.
*
 AC-YEAR.
        ACCEPT SUBS-YEAR LINE 08 POSITION 45 SIZE 4
                                               PROMPT TAB.
        IF NOT YEARR GO TO AC-YEAR.
*
 AC-MONTH.
        ACCEPT SUBS-MONTH LINE 09 POSITION 45 SIZE 2
                                               PROMPT TAB.
        IF NOT MONTHH   GO TO AC-MONTH.
*
 AC-DAY.
        ACCEPT SUBS-DAY  LINE 10 POSITION 45 SIZE 2
                                              PROMPT TAB.
        IF NOT DAYY    GO TO AC-DAY.
*
 AC-CODE.
        ACCEPT SUBS-CODE LINE 11 POSITION 45 PROMPT TAB.
        IF NOT CODEE GO TO AC-CODE.
*
 AC-AREA.
       ACCEPT SUBS-AREA LINE 12 POSITION 45 PROMPT TAB.
```

```
       IF NOT AREAA GO TO AC-AREA.
*
 AC-CNTR.
       ACCEPT SUBS-CNTR LINE 13 POSITION 45 PROMPT TAB.
       IF SUBS-CNTR NOT NUMERIC OR SUBS-CNTR = ZERO
                                          GO TO AC-CNTR.
*
 AC-LREAD.
       ACCEPT SUBS-LREAD LINE 14 POSITION 45 PROMPT TAB.
       IF SUBS-LREAD NOT NUMERIC OR SUBS-LREAD = ZERO
                                          GO TO AC-LREAD.
*
* كتابه السجل على القرص بعد ان تاكدنا من انه صحيح
*
       WRITE SUBS-REC
          INVALID KEY DISPLAY "السجل لم يضف الرجاء اعد العمل"
             LINE 23 POSITION 10
                 ACCEPT DUMMY GO TO EXT-INS.
       DISPLAY "الســـــــــــجل اضيف"
       LINE 23 POSITION 10 ACCEPT DUMMY.
 EXT-INS.
       EXIT.
*
* الفقره التاليه هى للاستفسار عن سجل دون امكانيه تعديله
*
 INQR-RTN.
           PERFORM ACCEPT-NO THRU EXT-AC.
           IF SUBS-NO = "00000" GO TO EXT-INQ.
           IF FLAG    = 1
              DISPLAY         "الســـجل غير موجود"
                              LINE 20 POSITION 10
                              ACCEPT  DUMMY GO TO EXT-UPD.
           PERFORM  DISPLAY-SCREEN.
           PERFORM  DISPLAY-FIELDS.
           DISPLAY
           "اضغط مفتاح الانتهاء للعوده الى الاستفسار من جديد"
           LINE     23 POSITION 5 ACCEPT DUMMY.
           GO TO INQR-RTN.
  EXT-INQ.
           EXIT.
*

*      الفقره التاليه هى لحذف سجل من الملف بتعليمه
* DELETE
*
 DELT-RTN.
       PERFORM ACCEPT-NO THRU EXT-AC.
       IF SUBS-NO = "00000" GO TO EXT-DEL.
       IF  FLAG = 1 DISPLAY
                "ليس هناك سجل لحذفه فضلا تاكد من الرقم"
                LINE 20 POSITION 10
                ACCEPT DUMMY GO TO EXT-DEL.
       DISPLAY SUBS-NAME LINE 20 POSITION 10 ERASE
           "هل انت متاكد من الحذف ، اجب بـ ن او لا"
                    LINE 22 POSITION 5.
       ACCEPT ANSR          LINE 21 POSITION 10 PROMPT TAB.
       IF      ANSR  = "ك" GO TO EXT-DEL.

* اذا كانت الاجابه ن اى نعم يتم الحذف كما فى التعليمه التاليه
*
       DELETE SUBSFL RECORD.
       DISPLAY "تم حذف السجل"  LINE 23 POSITION 30.
       ACCEPT DUMMY.
 EXT-DEL.
       EXIT.
```

```
*
* فقره قبول رقم السجل المراد معالجته ثم قراءه الملف
*                                  لمعرفه هل هو موجود ام لا ؟
 ACCEPT-NO.
       MOVE ZERO TO FLAG.
       DISPLAY
       "ادخل رقم المشترك المراد العمل عليه او اضغط              لانتهاء"
       LINE 10 POSITION 12 ERASE
       "RETURN"
       LINE 10 POSITION 55 SIZE 6.
       ACCEPT SUBS-NO LINE 11 POSITION 22 SIZE  5  PROMPT TAB.
       IF SUBS-NO = "00000" GO TO EXT-AC.
       READ SUBSFL KEY IS SUBS-NO
            INVALID KEY MOVE 1 TO FLAG GO TO EXT-AC.
       MOVE ZERO TO FLAG.
 EXT-AC.
       EXIT.
*
* الفقره التاليه هى لعرض  الشاشه ثم عرض السجل عليها *
*
 DISPLAY-SCREEN.
        DISPLAY
        "شـــــــــــــــــركه الكهرباء          "
        LINE 1 POSITION 25 ERASE
        "برنامج بناء وتعديل ملف المستهلكين الاساسى"
        LINE 2 POSITION 35
        " رقم المشترك 1-"     LINE 3  POSITION 25
        " اسم المشترك 2-"     LINE 5  POSITION 25
        "عنوان المشترك3-"     LINE 6  POSITION 25
        "تاريخ الاشتراك4-"    LINE 7  POSITION 25
        "السنه5-"             LINE 8  POSITION 25
        "الشهر6-"             LINE 9  POSITION 25
        "اليوم 7-"            LINE 10 POSITION 25
        "رمز  الاشتراك 8-"    LINE 11 POSITION 25
        "رمز   المنطقه 9-"    LINE 12 POSITION 25
        " رقم العداد10-"      LINE 13 POSITION 25
        " اخر قراءه 11-"      LINE 14 POSITION 25.
 EXT-SC.
       EXIT.
*
 DISPLAY-FIELDS.
       DISPLAY     SUBS-NO     LINE 3    POSITION 45
                   SUBS-NAME   LINE 5    POSITION 45
                   SUBS-ADRES  LINE 6    POSITION 45
                   SUBS-YEAR   LINE 8    POSITION 45
                   SUBS-MONTH  LINE 9    POSITION 45
                   SUBS-DAY    LINE 10   POSITION 45
                   SUBS-CODE   LINE 11   POSITION 45
                   SUBS-AREA   LINE 12   POSITION 45
                   SUBS-CNTR   LINE 13   POSITION 45
                   SUBS-LREAD  LINE 14   POSITION 45.
 EXT-FLD.
        EXIT.
```

١٣ ـ ٤ برنامج معالجة بيانات وطباعة :

البـرنـامـج الـتـالي يعالج ملفين أحدهما تسلسلي SEQUENTIAL والآخر مفهرس INDEXED SEQUENTIAL

يـقـرأ المـلـف الـتـسلسلي و يؤخذ المفتاح منه لقراءة الملف المفهرس، وعندما تكون الـقـراءة صحيحة VALID KEY تتم معالجة المطلوب وهو مثلاً حساب فاتورة المستهلك باستعمال حقول من الملفين وطباعة الناتج على الطابعة. ثم معاودة القراءة من جديد.

هـذا الأسلوب من المعالجة بسيط إذ يستخدم ثلاثة ملفات الملف المتسلسل والملف المفهرس وملف الطباعة لطباعة الناتج.

أحـيـانـاً يـطـلـب في البرنامج تعديل حقل ضمن سجل الملف المفهرس لذلك يحرك الحقل المرغوب (في مثالنا هنا مقدار القراءة الحالية إلى مقدار القراءة الأخيرة) من ملف الادخـال إلى المـلـف المـفـهـرس و يعاد كتابة السجل REWRITE وذلك لتعديل حقل أثناء المعالجة.

انظر البرنامج :

١٣ ـ ٤ ـ ١ نص التمرين :

لـديـك مـلـفـان : الأول واسمه INDXFL وهو ملف مفهرس يحتوي على معلومات عن المشتركين في شركة الكهرباء وشكل سجله :

	FIELD NAME	PICTURE
رقم المشترك	IDX-NO	X (5)
أســـــمه	IDX-NAME	X (20)
عنوانه	IDX-ADRESS	X (24)
تاريخ الأشتراك	IDX-DATE	9 (8)
رمز الأشتراك	IDX-CODE	99

رمز المنطقة	IDX-AREA	99
رقم العداد	IDX-CNTR	9 (7)
القراءة الأخيرة	SDX-LREAD	9 (7)
فـــراغ	BLANK	X (5)

والثاني واسمه NCONSMP وهو ملف متسلسل يحتوي على معلومات عن استهلاك المشتركين في شركة الكهرباء وشكل سجله :

	FIELD NAME	PICTURE
رقم المشترك	CONS-NO	9 (5)
مقدار القراءة الحالية	CONS-READ	9 (7)
تاريخ القراءة	CONS-DATE	9 (8)
ملاحظات	CONS-COMNT	X (30)
فـــراغ	FILLER	X (30)

المطلوب اقرأ ملف المستهلكين NCONSMP أولاً ثم حرك الحقل CONS-NO إلى مفتاح الملف INDXFL وهو IDX-NO واقرأ ملف المشتركين الأساسي بالمفتاح السابق وإذا لم يوجد هذا السجل أعد قراءة سجل جديد من ملف المستهلكين وإلاّ تابع العمل، كالتالي :

المطلوب طباعة قائمة بالمشتركين واستهلاكهم والمستحق عليهم علماً بأن الاستهلاك يحسب بالطريقة التالية :

١ - يطرح مقدرا القراءة السابقة IDX-LREAD من مقدار القراءة الحالية CONS-READ و يوضح الناتج في حقل اسمه W-READ وهويمثل الاستهلاك.

٢ - إذا كان رمز الاشتراك ١ أو كان الاستهلاك W-READ أقل من ١٠٠ كيلووات ساعي فيحسب سعر الكيلووات بـ ٥ هللات.

٣ - إذا كان رمز الاشتراك ٢ أو كان الاستهلاك W-READ أكبر من ٩٩ كيلووات

ساعي وأصغر من ٥٠٠ فيحسب سعر الكيلووات ساعي بـ ٤ هللات.

٤ - إذا كان رمز الاشتراك ٣ أو كان الاستهلاك W-READ أكبر من ٥٠٠ كيلووات ساعي فيحسب سعر الكيلووات بـ ٣ هللات.

٥ - إذا كان رمز الأشتراك من : ٠٤ إلى ٠٦ فيعفى من دفع قيمة الاستهلاك.

٦ - إذا كان رمز الأشتراك من ٠٧ ـــ ١٠ فيحسب سعر الكيلووات ساعي بـ ٢ هللة.

٧ - تضاف قيمة رسوم الجباية بمعدل ٥٠ هللة لكل مشترك مهما كان رمز اشتراكه أو مقدار استهلاكه.

٨ - تضاف قيمة ٢٪ من قيمة الفاتورة الاجمالية كرسم تحصيل.

٩ - تجمع كل القيم السابقة في حقل يسمى المستحق للدفع.

١٠ - ينقل مقدار القراءة الحالية CONS-READ إلى حقل IDX-READ الموجود في ملف المشتركين الأساسي و يعاد كتابة السجل في الملف الأساسي.

١١ - يطبع السجل كما هو وارد في التوصيف المرفق.

١٢ - يكرر العمل حتى نهاية ملف الاستهلاك.

١٣ - يطبع في نهاية العمل : عدد المستهلكين : عدد الكيلووات المصروفة : مجموع المبالغ المستحقة.

ملاحظات :

١ ـــ اطبع التاريخ ورقم الصفحة والعناوين برأس كل صفحة.

٢ ـــ اطبع في الصفحة الواحدة ٢٥ سطراً فقط.

٣ ـــ اطبع المجاميع على صفحة مستقلة.

٤ ـــ في حالة تضارب شرطين في حساب الفاتورة يؤخذ بالحالة الأقل تكلفة بالنسبة للمستهلك. أنظر صفحة رقم ٦٠٨ من أجل توصيف شكل المخرجات والصفحة رقم ٥٩٨ للنتائج

١٣ ـ ٤ ـ ٣ نص برنامج معالجة البيانات والطباعة

```
* هل تمرين معالجة بيانات وطباعتها ١٣-٤-٣ *
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT18.
 AUTHOR.      KHAYAT-JNAID.
*
*  يرى هذا التمرين الخطوات الاساسية فى كتابة
*  برنامج يعالج البيانات ثم يطبع النتائج.
*
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.    IBM-370.
   OBJECT-COMPUTER.    IBM-370.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
       SELECT INDXFL  ASSIGN  TO IDXFL
       ORGANIZATION  IS INDEXED
       ACCESS   MODE  IS RANDOM
       RECORD   KEY   IS IDX-NO.
       SELECT  NCONSMP ASSIGN TO UT-S-NCON.
       SELECT  HGRIDAT ASSIGN TO UT-S-HDAT.
       SELECT P-FILE  ASSIGN  TO UT-S-PRFL.
 DATA DIVISION.
 FILE  SECTION.
*   :  توصيف ملفات الادخال
*  .     INDXFL وهو الملف الاساسى المفهرس
*        والمطلوب معالجته وتعديله.
*
 FD     INDXFL     LABEL RECORD  IS STANDARD
                        DATA RECORD IS IDX-REC.
 01     IDX-REC.
        03    IDX-NO      PIC  X(5).
        03    IDX-NAME    PIC  X(20).
        03    IDX-ADRS    PIC  X(24).
        03    IDX-DATE    PIC  9(8).
        03    IDX-CODE    PIC  99.
        03    IDX-AREA    PIC  99.
        03    IDX-CNTR    PIC  9(7).
        03    IDX-LREAD   PIC  9(7).
        03    FILLER      PIC  X(5).
```

```
*
*     NCONSMP وهو ملف الحركه والذى يحتوى على                *
*     المعلومات الجديده المعدله (بكسر الدال)                 *
*
 FD      NCONSMP   LABEL RECORD  IS STANDARD
         BLOCK CONTAINS 0 RECORDS
                       DATA RECORD IS CON-REC.
 01      CON-REC.
         03    CON-NO     PIC  X(5).
         03    CON-READ   PIC  9(7).
         03    CON-DATE.
            05 CON-YEAR PIC  9999.
            05 CON-MONT PIC  99.
            05 CON-DAY  PIC  99.
         03    CON-CMNT  PIC  X(30).
         03    FILLER    PIC  X(30).
*
*     يحتوى على التاريخ الهجرى لصعوبه                        *
*     الحصول عليه من الحاسب
 FD      HGRIDAT   LABEL RECORD  IS OMITTED
         BLOCK CONTAINS 0 RECORDS
                          DATA RECORD IS H-REC.
 01      H-REC.
         05 H-MONTH  PIC  99.
         05 H-YEAR   PIC  9999.
         05 FILLER   PIC  X(74).
*
*     توصيف ملف الطباعه :
*
 FD      P-FILE     LABEL RECORD  IS OMITTED
         BLOCK CONTAINS 0  RECORDS.
*
*     يوصف غالبا سجل بطول ١٣٢ حرفا وهو عرض الطابعه          *
*WORKING-STORAGE وتحرك السجلات الاخرى الموصفه فى             *
*     اليه ويتم الكتابه باسم هذا السجل P-REC وما يحتوى       *
*     من بيانات مختلفه .
*
 01      P-REC          PIC      X(132).
*
 WORKING-STORAGE SECTION.
```

```
*-------------------------------------------------------*
*  توصف العناوين كما هو واضح في ورقة                      *
*  التوصيف المرفقة PRINT CHART وتسمى                      *
*  العناوين الثابتة او الفراغات                           *
*  بـ FILLER بينما الحقول المتغيرة                        *
*  مثل التاريخ ورقم الصفحة باسماء                         *
*  مناسبة تدل على محتوى الحقل لتسهيل                      *
*  معالجته ________________________________________________ *
*-------------------------------------------------------*
 01    HDR-1.
   03   FILLER    PIC   X(58)  VALUE SPACES.
   03   FILLER    PIC   X(16)  VALUE
    'شركة الكهرباء'.
   03   FILLER    PIC   X(58)  VALUE SPACES.
*
 01    HDR-2.
   03   FILLER    PIC   XX     VALUE '19'.
   03   YEAR-P    PIC   99.
   03   FILLER    PIC   X      VALUE '/'.
   03   MONTH-P   PIC   99.
   03   FILLER    PIC   X      VALUE '/'.
   03   DAY-P     PIC   99.
   03   FILLER    PIC   X(08)  VALUE 'تاريخ'.
   03   FILLER    PIC   X(35)  VALUE SPACES.
   03   FILLER    PIC   X(26)  VALUE
        'قائمة بالمشتركين واستهلاكهم'.
   03   FILLER    PIC   X(39)  VALUE SPACES.
   03   P-CTR     PIC   ZZ9.
   03   FILLER    PIC   X(10)  VALUE
        'صفحة رقم :'.
   03   FILLER    PIC   X(10)  VALUE SPACES.
*
 01    HDR-3.
   03   FILLER    PIC   X(60)  VALUE SPACES.
   03   HDYEAR    PIC   9999.
   03   FILLER    PIC   X(05)  VALUE ' عام'.
   03   HDMONT    PIC   99.
   03   FILLER    PIC   X(62)  VALUE 'الشهر'.
*
 01    HDR-4.
   03   FILLER    PIC   X(20)  VALUE
```

```
           '       المستحق للدفع     '.
   03     FILLER    PIC   X(22) VALUE
           'العداد    مقدار   الاستهلاك'.
   03     FILLER    PIC   X(28) VALUE
           'رمز   المنطقة  رمز  الاستهلاك  رقم '.
   03     FILLER    PIC   X(20) VALUE
           'عنوان  المشترك         '.
   03     FILLER    PIC   X(21) VALUE
           'اسم  المشترك             '.
   03     FILLER    PIC   X(19) VALUE
           'رقم  المشترك           '.
*-----------------------------------------------------------------*
*  السطر  التالي هو لتوصيف سطر جسم  التقرير                        *
*  BODY LINE  اي  السطر  الذي يحتوي على                             *
*  البيانات  المتغيرة  والذي  سيتكرر على  طول                       *
*  الصفحة                                                          *
*-----------------------------------------------------------------*
 01       DATA-LINE.
   03     FILLER    PIC   XXX      VALUE SPACES.
   03     P-PAY     PIC   Z(6)9.99.
   03     FILLER    PIC   X(11) VALUE SPACES.
   03     P-CONSMP  PIC   Z(5)99.
   03     FILLER    PIC   X(07) VALUE SPACES.
   03     P-CONTR   PIC   Z(5)99.
   03     FILLER    PIC   X(09) VALUE SPACES.
   03     P-CODE    PIC   99.
   03     FILLER    PIC   X(09) VALUE SPACES.
   03     P-AREA    PIC   99.
   03     FILLER    PIC   X(05) VALUE SPACES.
   03     P-ADRESS  PIC   X(24).
   03     FILLER    PIC   XX       VALUE SPACES.
   03     P-NAME    PIC   X(20).
   03     FILLER    PIC   X(05) VALUE SPACES.
   03     P-NO      PIC   X(5).
   03     FILLER    PIC   X(8)     VALUE SPACES.
*
 01       FIN-1.
   03     FILLER    PIC   X(58) VALUE SPACES.
   03     FILLER    PIC   X(16) VALUE
           'الخلاصة  النهائية'.
   03     FILLER    PIC   X(58) VALUE SPACES.
```

```
 01     FIN-2.
  03    FILLER    PIC  X(50) VALUE SPACES.
  03    PCNTR1    PIC  ZZZ999.
  03    FILLER    PIC  X(68) VALUE
        '                 : عدد المستهلكين'.
 01     FIN-3.
  03    FILLER    PIC  X(50) VALUE SPACES.
  03    PCNTR2    PIC  Z(7)99.
  03    FILLER    PIC  X(64) VALUE
        ': مقدار الكيلووات المصروفة'.
 01     FIN-4.
  03    FILLER    PIC  X(50) VALUE SPACES
  03    PCNTR3    PIC  Z(7)9.99.
  03    FILLER    PIC  X(62) VALUE
        ': مقدار المبالغ المستحقة'.
*
*
*       مساحة لقبول التاريخ
 01     CUR-DATE.
  03    C-YEAR    PIC  99.
  03    C-MONTH   PIC  99.
  03    C-DAY     PIC  99.
*
 77     F-ST      PIC     99        VALUE ZERO.
*       عدادات تستخدم لتجميع
*       عدد الاسطر وعدد الصفحات *
 77     LN-CTR    PIC     99        VALUE 30.
 77     P-CNTR    PIC     999       VALUE ZERO.
*       عدادات تستخدم كمساحات عمل وسيطة *
 77     W-READ    PIC     9(7).
 77     W-PAY     PIC     9(5)V99.
 77     HALALAH   PIC     9V99.
*       عدادات تستخدم لتجميع
*       المبالغ او المقادير المصروفة *
 77     CNTR1     PIC     9(6)      VALUE ZERO.
 77     CNTR2     PIC     9(9)      VALUE ZERO.
 77     CNTR3     PIC     9(8)V99 VALUE ZERO.
```

```
*
 PROCEDURE DIVISION.
*
* الخطوه الاولى هى فتح الملفات *
       OPEN        INPUT   NCONSMP HGRIDAT
                   I-O     INDXFL
                   OUTPUT  P-FILE.
*
* قبول التاريخ من الحاسب وتحريكه *
* الى حقول الطباعه ثم طباعه
* العناوين فى الفقره NEW-PAGE,
*
       ACCEPT    CUR-DATE        FROM DATE.
       PERFORM   INSP            THRU EXTS.
       MOVE      SPACE           TO   P-REC.
       WRITE     P-REC           AFTER PAGE.
       MOVE      C-YEAR          TO    YEAR-P.
       MOVE      C-MONTH         TO    MONTH-P.
       MOVE      C-DAY           TO    DAY-P.
       READ     HGRIDAT.
       MOVE      H-MONTH         TO    HDMONT.
       MOVE      H-YEAR          TO    HDYEAR.
       CLOSE    HGRIDAT.
       PERFORM   NEW-PAGE.
*
 MAIN-LOOP.
*        القراءه من ملف التعديل اولا
       READ    NCONSMP   AT END
                   PERFORM FINAL-RTN
                     CLOSE NCONSMP INDXFL
                           P-FILE    STOP RUN.
*
*   ثم تحريك رقم المشترك من ملف الاستهلاك
*   الى حقل رقم المشترك فى الملف الاساسى
*   والقراءه به عشوائيا RANDOMLY حسب قيمه
*   المفتاح فاذا تمت القراءه اى السجل
*   موجود فينتقل التنفيذ للتعليمه PERFORM
*   والا تتشكل حالة INVALID KEY ويذهب
*   التنفيذ الى الفقره MAIN-LOOP للقراءه
*   من جديـــــــــــــــــــــــــــــد
*
       MOVE    CON-NO       TO      IDX-NO.
```

```
          READ   INDXFL  INVALID KEY
                              GO TO  MAIN-LOOP.
          PERFORM  CALCULATE-RTN.
          GO  TO  MAIN-LOOP.
*
* الفقرة التالية هي لحساب استهلاك مشترك *
*
 CALCULATE-RTN.
          MOVE   ZERO        TO    HALALAH.
*
*    حساب الإستهلاك بالكيلو وات
*
          SUBTRACT IDX-LREAD FROM  CON-READ
                                    GIVING W-READ.
*
*    تصنيف المستهلك تبعا لإستهلاكه او
*    الرمز اشتراكه, ثم وضع قيمة استهلاك
*    الوحدة في حقل HALALAH لحساب مجمل
*    الإستهلاك.
*
          IF  IDX-CODE = 01 OR W-READ < 100
                      MOVE  0.05  TO  HALALAH.
*
          IF  IDX-CODE = 02 OR
             ( W-READ > 99  AND W-READ  < 500 )
                      MOVE  0.04  TO  HALALAH.
*
          IF   IDX-CODE = 03 OR W-READ > 500
                      MOVE  0.03  TO  HALALAH.
*
          IF IDX-CODE > 06  AND IDX-CODE < 11
                      MOVE  0.02  TO  HALALAH.
*
          IF IDX-CODE > 05  AND IDX-CODE < 07
                      MOVE  ZEROS TO  HALALAH.
*
*         حساب التكلفة
*
          MULTIPLY  W-READ  BY  HALALAH
                                GIVING W-PAY.
          ADD  0.50   TO   W-PAY.
```

```
*
*  حساب ٢٠% من التكلفه واضافه الناتج الى
*  التكلفه مباشره (اى نضرب ب ١٢٠% ).
*
*
       MULTIPLY 1.20 BY W-PAY.
*
*
*  تحريك حقل اخر قراءه CON-READ من ملف
*  استهلاك الى الملف الاساسى واعاده كتابه
*  السجل بعد تعديله.
*
       MOVE      CON-READ TO      IDX-LREAD.
       REWRITE   IDX-REC.
*                طباعه السجل على الطابعه
       PERFORM   PRINT-RTN.
*
*  تجميع عدد المشتركين والاستهلاك والمبالغ
*  المستحقه فى عدادتها المناسبه
*
        ADD       1          TO     CNTR1.
        ADD       W-READ     TO     CNTR2.
        ADD       W-PAY      TO     CNTR3.
   EXT.
        EXIT.

*
*
*  تحريك الحقول من ملفات الادخال, وكذلك
*  DATA-LINE الحقول الحسابيه الوسيطه الى
*  الموصف فى قسم توصيف البيانات الداخليه
*
 PRINT-RTN.
        MOVE    IDX-NO       TO     P-NO.
        MOVE    IDX-NAME     TO     P-NAME.
        MOVE    IDX-ADRS     TO     P-ADRESS.
        MOVE    IDX-CODE     TO     P-CODE.
        MOVE    IDX-AREA     TO     P-AREA.
        MOVE    IDX-CNTR     TO     P-CONTR.
        MOVE      W-READ     TO     P-CONSMP.
        MOVE      W-PAY      TO     P-PAY.
```

```
*
*
*   :  كتابه السجل على ملف الطباعه
*
       MOVE       DATA-LINE TO P-REC.
       PERFORM   INSP          THRU EXTS.
       WRITE    P-REC
                     AFTER ADVANCING 2 LINES.
*
*
*   اضافه 1 لعداد الاسطرLN-CTR ثم مقارنه
*   محتواه مع العدد 25 لان عدد الاسطر
*   المسموح طباعتها فى الصفحه الواحده
*   هو 24 , فاذا كان اكبر تقلب الصفحه
*   وتطبع العناوين على الصفحه الجديده
*   ثم يذهب التنفيذ الى قراءه سجل جديد ,
*   واذا لم يكن اكبر فيذهب التنفيذ الى
*   معالجه سجل جديد مباشره .
*
       ADD        1            TO       LN-CTR.
       IF         LN-CTR       >        25
                               PERFORM    NEW-PAGE.
 EXTL.
       EXIT.
*
*
*   فقره قلب الصفحه وكتابه العناوين:
*   ( لاحظ اضافه 1 لعداد الصفحات PG-CTR
*   وتحريك القيمه صفر لعداد الاسطر )
 NEW-PAGE.
       MOVE       ZERO         TO       LN-CTR.
       ADD        1            TO       P-CNTR.
       MOVE       P-CNTR       TO       P-CTR.
       MOVE       SPACES       TO       P-REC.
       WRITE      P-REC        AFTER    PAGE.
       MOVE        HDR-1       TO       P-REC.
       PERFORM     INSP             THRU EXTS.
       WRITE      P-REC       AFTER 1 LINE.
       MOVE        HDR-2       TO       P-REC.
       PERFORM     INSP             THRU EXTS.
       WRITE      P-REC       AFTER 2 LINE.
       MOVE        HDR-3       TO       P-REC.
       PERFORM     INSP             THRU EXTS.
```

```
       WRITE    P-REC      AFTER 1 LINE.
       MOVE      HDR-4      TO      P-REC.
       PERFORM   INSP            THRU EXTS.
       WRITE    P-REC      AFTER 2 LINE.
   EXTP.
       EXIT.
*          الفقرة التالية هي لطباعة المجاميع
   FINAL-RTN.
        MOVE SPACES TO P-REC.
        WRITE   P-REC  AFTER ADVANCING PAGE.
        MOVE    SPACES TO     P-REC.
        MOVE      FIN-1 TO     P-REC.
       PERFORM   INSP            THRU EXTS.
        WRITE   P-REC  AFTER 30.
        MOVE    CNTR1   TO     PCNTR1.
        MOVE      FIN-2 TO     P-REC.
       PERFORM   INSP            THRU EXTS.
        WRITE   P-REC  AFTER 2.
        MOVE    CNTR2   TO     PCNTR2.
        MOVE      FIN-3 TO     P-REC.
       PERFORM   INSP            THRU EXTS.
        WRITE   P-REC  AFTER 2.
        MOVE    CNTR3   TO     PCNTR3.
        MOVE      FIN-4 TO     P-REC.
       PERFORM   INSP            THRU EXTS.
        WRITE   P-REC  AFTER 2.
   EXTF.
       EXIT.
*      يتم تحويل الارقام من لاتيني الى عربي *
  INSP.
       INSPECT P-REC REPLACING ALL '.' BY ','
       INSPECT P-REC REPLACING ALL '0' BY '٠'
       INSPECT P-REC REPLACING ALL '1' BY '١'
       INSPECT P-REC REPLACING ALL '2' BY '٢'
       INSPECT P-REC REPLACING ALL '3' BY '٣'
       INSPECT P-REC REPLACING ALL '4' BY '٤'
       INSPECT P-REC REPLACING ALL '5' BY '٥'
       INSPECT P-REC REPLACING ALL '6' BY '٦'
       INSPECT P-REC REPLACING ALL '7' BY '٧'
       INSPECT P-REC REPLACING ALL '8' BY '٨'
       INSPECT P-REC REPLACING ALL '9' BY '٩'.
  EXTS.
        EXIT.
```

١٣ ــ ٥ تمارين عامة :

تمــرين رقم (١) :

اكـتـب برنامجاً بلغة كوبول يطبع في يسار سطر الطباعة الأرقام من ١ ــ ١٠ بمعدل رقم في كل سطر، وضمن خمسة مجموعات، واترك سطر فراغ بين كل مجموعتين.

ــ استخدام تعليمة PERFORM ــ

شكل الاخراج :

```
ϕ1
ϕ2
ϕ3
ϕ4      مجموعة ــ ١
ϕ5
ϕ6
ϕ7
ϕ8
ϕ9
1ϕ

ϕ1
ϕ2
ϕ3
ϕ4      مجموعة ــ ٢
```

```
05
06
07
08
09
10
01
..
..
..
..      مجموعة ـــ ٣
..
..
```

تمـــرين رقم (٢) :

اكـتـب برنامجاً بلغة كوبول بحيث يقرأ سجل زبون ــ عميل ــ CUSTOMER في بـنـك تجـاري ويحـسـب لـه الرصيد الجديد؛ علماً بأن أسماء الزبائن موجودة على ملف أسـاسي مـتسلسل SEQUENTIAL على قرص ممغنط ، واسم الملف DEPFILE ــ انظر توصيف الملف في الصفحة التالية :

مع مراعاة ما يلي :

١ ــ بـعـد قـراءة كـل سـجل تحقق من حقل رمز السجل D-CODE فاذا كان هذا الحقل يحتوي على الحرف 'D' فهذا يعني أن السجل ملغى فتجاهله ولا تعالجه .

٢ ــ أمـا إذا كـان الحـقـل D-CODE يحـتـوي على الرمز 'A' فهذا يعني أن السجل

تحت المعالجة ACTIVE فأجر عليه حساب الرصيد الجديد وذلك حسب الخطوات التالية :

NEW-BAL الرصيد الجديد = رصيد آخر المدة D-ENDBAL + المبلغ المودع لتاريخه D-DPST — المبلغ المسحوب لتاريخه D-DRFT

NEW-BAL = D-ENDBAL + D-DPST—D - DRFT

بعد اجراء عملية الحساب السابقة اطبع محتوى السجل مع الرصيد الجديد على الطابعة وكما هو مبين في النموذج المرفق الخاص بالطباعة .

٣ ــ كرر العمل السابق من أجل كل سجل .

٤ ــ في نهاية العمل اطبع :

مجموع المودعين ــ عدا المحذوفين

مجموع الأ رصدة القديمة

مجموع المبالغ المودعة

مجموع المبالغ المسحوبة

مجموع الأ رصدة الجديدة

مع مراعاة ما يلي :

١ ــ اطبع التاريخ ورقم الصفحة برأس كل صفحة .

٢ ــ اترك فراغاً بين كل سطري كتابة .

٣ ــ لا تطبع في الصفحة الواحدة أكثر من ٢٢ سطر .

٤ ــ اطبع المجاميع النهائية على صفحة مستقلة .

٥ ــ استعمل الفاصلة (,) بدلاً من (.) للدلالة على الفاصلة العشرية .

أظهر الاشارة الجبرية للرصيد الجديد على يسار حقل طباعة هذا الحقل .

توصيف ملف الادخال DEPFILE :

	FIELD NAME	PICTURE	
رمز السجل	D-CODE	X	
رقم الحساب	D-ACNO	9 (5)	
اسم الزبون	D-ACNAME	X (3ϕ)	
رصيد آخر المدة	D-ENDBAL	9 (7) V99	أنظر الصفحة رقم
المبلغ المودع لتاريخه	D-DPST	9 (7) V99	٦٠٩ لتوصيف شكل
المبلغ المسحوب لتاريخه	D-DRFT	9 (7) V99	المخرجات والصفحة
فـــــراغ	FILLER	X (17)	رقم ٦٠١ للنتائج.

تمـــرين رقم (٣) :

لديك ملف اسمه VALFILE وهو ملف متسلسل SEQUENTIAL موجود على قرص ممغنط ويحتوي على أسماء وأرقام ورمز المنطقة بالاضافة إلى معلومات أخرى.

المطلوب كتابة برنامج يقرأ هذا الملف و يعالجه، وفي نهاية العمل اطبع جدولاً تكرارياً يحتوي على رمز المنطقة واسم المنطقة وعدد المشتركين في كل منطقة والمجموع الكلي للمشتركين. حسب الشكل التالي :

الرمـــز	اســـــم المنطقـــــة	عدد المشتركين
Z9	XXXXXXX	ZZ9
Z9	XXXXXXX	ZZ9
		.
		.
		.
		.
		.
		.
**	العـــــدد الـــــكلي	ZZ99

علماً بأن توصيف السجل كالتالي :

	FIELD NAME	PICTURE
	VAL-NO	9 (5)
	VAL-NAME	X (3ϕ)
	VAL-ADRES	X (24)
	VAL-DATE	9 (8)
	VAL-CODE	99
رمز المنطقة	VAL-AREA	99
	BLNK	9(8)

وأن رموز المناطق كالتالي :

الرمـــز	اســـم المنطقة
٠١	الرياض
٠٢	جـــدة
٠٣	الدمام
٠٤	القصيم
٠٥	حـــائل
٠٦	المدينة
٠٧	أبهـــا
٠٨	بلجرشي
٠٩	عســـير
١٠	شـــقرة

تمـرين رقم (٤) :

تـريـد شـركـة صـنـاعية احتساب الأجر الاضافي لعمالها، علماً بأن جميع سجلات RECORDS العـمال موجودة على ملف مفهرس INDEXED FILE اسمه PAYMST واسـم الـسجـل فيـه RECMST ومـوجـود على قـرص ممـغـنـط DISC وشـكـل السجل فيه كالتالي :

	FIELD NAME	PICTURE
رقم العامل	(KEY).... EMP-NO	X (5)
اسم العامل	EMP-NAME	X (3ϕ)
أجره الشهري	EMP-SALARY	9 (5) V 99
	EMP-HORS1	999
	EMP-HORS2	999
	EMP-HORS3	999
مجموع الأجر الاضافي	EMP-OVTIME	9 (5) V99
	EMP-FLER	X (22)

حيث أن :

- EMP-SALARY : يمثـل الأجـر الشهري للعامل بمعدل ٨ ساعات عمل يومياً في ٣٠ يوم للشهر.
- EMP-HORS1 : تمـثـل عـدد الـسـاعات التي تحسب أجرتها وكأنها ساعة عمل واحدة لكل ساعة.
- EMP-HORS2 : تمـثل عدد الساعات التي تحسب أجرتها وكأنها ساعة ونصف عمل عادي لكل ساعة.

— EMP-HORS3 : تمثل عدد الساعات التي تحسب أجرتها وكأنها ساعتي عمل عادي لكل ساعة.

— EMP-OVTIME : مبلغ الأجر الاضافي المطلوب حسابه.

والمطلوب ما يلي :

كتابة برنامج يحسب الأجر الاضافي EMP-OVTIME لكل عامل و يكتبه على نفس السجل بشرط الاّ يتجاوز الأجر الاضافي مقدار نصف الأجر الشهري :

و يتم العمل وفق الخطوات التالية :

١ ـــ يقرأ الملف تسلسلياً.

٢ ـــ يجري حساب أجرة الساعة الواحدة.

٣ ـــ تضرب أجرة الساعة الواحدة بعدد ساعات العمل الاضافي على التوالي ولثلا ثة الأنواع السابقة:

EMP-HORS3 , EMP-HORS2 , EMP-HORS1

٤ ـــ يقارن الناتج مع مقدار نصف الراتب :

— فاذا كان الناتج أصغر أو مساوٍ لنصف الراتب ينقل الناتج إلى حقل EMP-OVTIME

— أما إذا كان أكبر من نصف الراتب فينقل مقدار نصف الراتب إلى حقل EMP-OVTIME واطبع على الطابعة رقم الموظف واسمه وراتبه الأساسي وعبارة «تجاوز الحد الأعلى».

٥ ـــ اعد كتابة السجل على القرص.

٦ ـــ كرر العمل من جديد.

٧ ــ في نهاية العمل أطبع :

مجموع الأجور الشهرية للعمال : 9999999.99

مجموع بدلات خارج الدوام : 9999999.99

عدد العمال الكلي : 9999

٨ ــ انهي العمل. أنظر الصفحة رقم ٦١٠ لتوصيف شكل المخرجات.

تمــرين رقم (٥) :

لديك ملفان متسلسلان ومفروزان تصاعدياً على نفس الحقل وهو رقم المشترك.

الملف الأول واسمه SUBFILE ملف المشتركين وشكل سجله :

	FIELD NAME	PICTURE
رقم المشترك	SUB-NO	9 (5)
اسمه	SUB-NAME	X (2∅)
عنوانه	SUB-ADRESS	X (24)
تاريخ الأشتراك	SUB-DATE	9 (8)
رمز الأشتراك	SUB-CODE	99
رمز المنطقة	SUB-AREA	99
رقم العداد	SUB-CNTR	9 (7)
مقدار آخر قراءة	SUB-LREAD	9 (7)
فــراغ	SUB-BLNK	X (5)

والملف الثاني واسمه CONSMPFL وهو ملف الاستهلاك وشكل سجله :

	FIELD NAME	PICTURE
رقم المشترك	CONS-NO	9 (5)
مقدار القراءة الحالية	CONS-READ	9 (7)

تاريخ القراءة	CONS-DATE	9 (8)
ملاحظات	CONS-COMNT	X (3ϕ)
فــراغ	FILLER	X (3ϕ)

المطلوب قراءة الملفين السابقين والتأكد أنه لكل سجل في ملف الاستهلاك CONSMPFL هناك سجل مقابل في ملف المشتركين الأساسي SUBFILE وفي حال وجود سجل أساسي انقل سجل الاستهلاك إلى ملف آخر اسمه NCONSMP وشكل سجله مطابق لسجل ملف الاستهلاك، أما إذا كان السجل في ملف الاستهلاك لا يحتوي مقابل له في سجل المشتركين اطبع محتوى هذا السجل على الطابعة ــ حسب الشكل المرفق ــ ولا تنقله على ملف الاستهلاك الجديد NCONSMP.

انهي العمل عند الانتهاء من ملف الاستهلاك. أنظرشكل المخرجات. صفحة ٦١١

تمــرين رقم (٦) :

لـدينا ملف على شريط ممغنط اسمه STDFILE يحتوي على أرقام و بيانات طلاب احدى الجامعات في المملكة ومفروز تصاعدياً حسب رمز الكلية ورقم السنة ضمن كل كلية.

المطلوب : كتـابـة برنامج يطبع سجل كل طالب ضمن كل سنة للكلية الواحدة وايجاد:

١ ــ مجموع طلاب كل سنة ضمن كل كلية MINOR TOTAL

٢ ــ مجموع طلاب كل كلية ضمن الجامعة MAJOR TOTAL

٣ ــ مجموع طلاب الجامعة OVERALL TOTAL

* لا تكتب في الصفحة أكثر من ٣٠ سطراً.

** عند طباعة المجاميع اترك سطرين فارغين قبل و بعد سطر المجاميع.

*** انظر الصفحة المرفقة من أجل توصيف الطباعة.

شكل سجل الملف STDFILE :

	FIELD NAME	PICTURE
رقم الطالب	STD-NO	9 (5)
اسم الطالب	STD-NAME	X (3ϕ)
رمز الكلية	STD-CLG	99
رقم السنة الدراسية	STD-YEAR	99
معلومات	STD-DATA	X (11)

مفتاح الحـــل :

١ ــ عين عدادات COUNTERS من أجل المجاميع .

٢ ــ عين مـسـاحـات عـمل WORK AREAS من أجل مقارنة رمز الكلية أو السنة مع رمز الكلية أو السنة للسجل السابق .

أنظر الصفحة رقم ٦١٢ لتوصيف شكل المخرجات.

الفصل الرابع عشر

المراجـــع

١٤ - ١ - المراجع العربية :

١ - **المعجم العربي الموحد لمصطلحات الحاسبات الألكترونية**، المنظمة العربية للعلوم الادارية ـــ جامعة الدول العربية، عمان ١٩٨١م.

٢ - **الحاسبات الألكترونية وسيلة لتطوير المجتمع**، د. أمين الصالح، وزارة الثقافة، دمشق ١٩٨١م.

٣ - **المبادىء الأساسية في الحاسبات الألكترونية**، السيد محمد السيد، دار المعارف، القاهرة ١٩٧٣م.

٤ - **أربعة مداخل للحاسبات الألكترونية**، أحمد المطاوع، عمان ١٩٧٨م.

٥ - **قاموس شرح المصطلحات العلمية للأجهزة المحاسبة الآلية**، د. حسام البيلاوي، منشأة المعارف، الاسكندرية ١٩٧٦م.

١٤ - ٢ - المراجع الأجنبية :

1. Davis, W.S. and Fisher, R.H. **"An Introduction to Structured Logic and Modular Program Design"**, Addison. Wesley, 1979.

2. Grauer, Robert T. **"A Cobol Book of Practice and Reference"**, Prentice - Hall, N.J., 1981.

3. Grauer, Robert T. **"Cobol a Vehicle for Information Systems"**, Prentice - Hall, N.J., 1981.

4. Grauer, Robert T. and Crawford, Marshal A. **"Structured Cobol, a Pragmatic Approach"**, Prentice - Hall N.J., 1981.

5. Harrison, W.S. **"A Programmer's Guide to Cobol"**, V.N.R., 1980.

6. McClure Carma **"Reducing Cobol Complexity Through Structured Programming"**, V.N.R., 1978.

7. N.C.C. **"Cobol Reference Summary"** 1980 Manchester, 1980.

8. N.C.C. **"Cobol : Student Notes and Work Book"**, Manchster, 1980.

9. N.C.C. **"Introducing Systems' Analysis and Design"**, Vol. 1 and 2, 1980.

10. N.C.R. Corp. **"NCR IMOS Cobol : Student Text"**, 4th Edition, Ohio, 1981.

11. IBM Corp. **"IBM OS VS Cobol compiler and Library, Programmer's Guide"** SC 28-6483-2 Oct. 1981.

12. IBM Corp. **"American National Standard Cobol, Coding, Illustration's"**, Feb., 1978.

13. IBM Corp. **"American National Standard Cobol Coding : Coding Techniques and Disk Applications Text"**, May, 1978.

14. IBM Corp. **"Time Sharing Option, 3270 Display Support and Structured Programming Facilities"**, V. 2.2, Program Ref. Manual, SH 20-1975-2 July, 1979.

15. Parkin, Arnold **"Cobol for Students"**, Arnold, 1975.

16. Philippakis A.S. and Kazmier L.J. **"Advanced Cobol"**, McGraw Hill, 1982.

17. Popkin, Gary S. **"Introductory Structured Cobol Programming"**, V.N.R. N.Y., 1981.

18. Philippakis, A.S. and Kazmier, L.J. **"Structured Cobol"**, McGraw-Hill 1981.

19. Shelly, G.B. and Cashman, T.J. **"Dos Job Control for Cobol Programmers"** Anheim, 1977.

20. Shelly, G.B. and Cashman, T.J. **"Introduction to Flowcharting and Computer**

Programming Logic", Anheim, 1978, 9th ed.

21. Silver and Silver **"Computer Algorithms and Flowcharting"** McGraw-Hill, 1975.

22. Sordillo Donald A. **"The Programmer's ANSI Cobol Reference Manual"**, Prentice - Hall, N.J., 1978.

23. Stern/Stern **"Structured Cobol Programming"**, John Wiley, 1980, 3rd ed.

24. Stuart, F. **"Introduction to Standard Cobol Programming"**, H.B.J. 1974.

25. Texas Instruments, **"Model 990 Computer Cobol Refernce Manual"**, 1980.

26. IBM Corp., **'OS/VS2 MVS JCL'**, GC28/0692/4, May 1979.

الفصل الخامس عشر

ملحقـــــــــات

١٥ ـــ ١ جدول رموز الاخراج *

Second Symbol \ First Symbol		Nonfloating Insertion Symbols									Floating Insertion Symbols						Other Symbols					
		B	0	/	,	.	$\{+,-\}^1$	$\{+,-\}^2$	{CR, DB}	cs	$\{Z, \bullet\}^1$	$\{Z, \bullet\}^2$	$\{+,-\}^1$	$\{+,-\}^2$	cs^1	cs^2	9	A X	S	V	P^1	P^2
Nonfloating Insertion Symbols	B	x	x	x	x	x	x			x	x	x	x	x	x	x	x	x		x		x
	0	x	x	x	x	x	x			x	x	x	x	x	x	x	x	x		x		x
	/	x	x	x	x	x	x			x	x	x	x	x	x	x	x	x		x		x
	,	x	x	x	x	x	x			x	x	x	x	x	x	x	x			x		x
	.	x	x	x	x		x			x	x		x		x		x					
	$\{+,-\}^1$																					
	$\{+,-\}^2$	x	x	x	x	x				x	x	x			x	x	x			x	x	x
	{CR, DB}	x	x	x	x	x				x	x	x			x	x	x			x	x	x
	cs						x															
Floating Insertion Symbols	$\{Z, \bullet\}^1$	x	x	x	x		x			x	x											
	$\{Z, \bullet\}^2$	x	x	x	x	x	x			x	x	x								x		x
	$\{+,-\}^1$	x	x	x	x					x			x									
	$\{+,-\}^2$	x	x	x	x	x				x			x	x						x		x
	cs^1	x	x	x	x		x								x							
	cs^2	x	x	x	x	x	x								x	x				x		x
Other Symbols	9	x	x	x	x	x	x			x	x		x		x		x	x	x	x		x
	A X	x	x	x													x	x				
	S																					
	V	x	x	x	x		x			x	x		x		x		x		x		x	
	P^1	x	x	x	x		x			x	x		x		x		x		x		x	
	P^2						x			x									x	x		x

[1] When character is to left of decimal point.
[2] When character is to right of decimal point.

* Sordillo P. 202

١٥ ـ ٢ الكلمات المحجوزة في لغة كوبول *

*COBOL RESERVED WORDS

ACCEPT
ACCESS
*ACTUAL
ADD
ADDRESS
ADVANCING
AFTER
ALPHABET
ALL
ALPHABETIC
ALPHANUMERIC
ALPHANUMERIC-EDITED
ALSO
ALTER
ALTERNATE
AND
ANY
*APPLY
ARE
AREA
AREAS
ASCENDING
ASSIGN
AT
AUTHOR

*BASIS
BEFORE
*BEGINNING
BIT
BITS
BOOLEAN
BOTTOM
BY

CALL
*CANCEL
CBL
CD
CF
CH
*CHANGED
CHARACTER
CHARACTERS
CLOCK-UNITS
CLOSE
COBOL
CODE
CODE-SET
COLUMN
*COM-REG
COMMA
COMP
*COMP-1
*COMP-2
*COMP-3
*COMP-4
COMPUTATIONAL
*COMPUTATIONAL-1
*COMPUTATIONAL-2
*COMPUTATIONAL-3
*COMPUTATIONAL-4

CONNECT
*CONSOLE
CONTAINS
CONVERSION
CONVERTING
CONTROL
CONTROLS
COPY
*CORE-INDEX
CORR
CORRESPONDING
COUNT
*CSP
CURRENCY
*CURRENT-DATE
*CYL-INDEX
*CYL-OVERFLOW
*C01
*C02
*C03
*C04
*C05
*C06
*C07
*C08
*C09
*C10
*C11
*C12

DATA

* Structured Cobol, Gramer and Crowford. PP 366-369

BLANK
BLOCK
DATE-WRITTEN
*DAY
DAY-OF-WEEK
DB-CONFLICT
DB-EXCEPTION
DB-KEY
DB-PRIVATE-KEY
DB-REALM-NAME
DB-RECORD-NAME
DB-SET-NAME
DB-STATUS
DE
*DEBUG
DEBUG-CONTENTS
DEBUG-ITEM
DEBUG-LENGTH
DEBUG-LINE
DEBUG-NAME
DEBUG-NUMERIC-
CONTENTS
DEBUG-SIZE
DEBUG-START
DEBUG-SUB
DEBUG-SUB-ITEM
DEBUG-SUB-N
DEBUG-SUB-NUM
DEBUG-SUB-1
DEBUG-SUB-2
DEBUG-SUB-3
DEBUGGING
DECIMAL-POINT
DECLARATIVES
DELETE
DELIMITED
DELIMITER
DEPENDING
DEPTH
DESCENDING
DESTINATION
DETAIL
DISABLE
*DISP
DISPLAY
*DISPLAY-ST

COMPUTE
CONFIGURATION
ENTER
*ENTRY
ENVIRONMENT
EOP
EQUAL
EQUALS
ERASE
ERROR
ESI
EVERY
EXCEEDS
EXCEPTION
EXCLUSIVE
*EXHIBIT
EXIT
EXTEND
*EXTENDED-SEARCH

FD
FILE
FILE-CONTROL
FILES
FILLER
FINAL
FIND
FINISH
FIRST
FOOTING
FOR
FREE
FROM

GENERATE
GET
GIVING
GO
*GOBACK
GREATER
GROUP

HEADING
HIGH-VALUE
HIGH-VALUES

*DATE
DATE-COMPILED
INPUT-OUTPUT
*INSERT
INSPECT
INSTALLATION
INTO
INVALID
IS

JUST
JUSTIFIED

KEEP
KEY

LABEL
*LABEL-RETURN
LAST
LEADING
*LEAVE
LEFT
LENGTH
LESS
LIMIT
LIMITS
LINAGE
LINAGE-COUNTER
LINE
LINE-COUNTER
LINES
LINKAGE
LOCALLY
LOCK
LOW-VALUE
LOW-VALUES

*MASTER-INDEX
MEMBER
MEMBERS
MEMBERSHIP
MEMORY
MERGE
MESSAGE
MODE
MODIFY

DIVIDE
DIVISION
DOWN
DUPLICATE
DUPLICATES
DYNAMIC

EGI
*EJECT
ELSE
EMI
ENABLE
END
END-OF-PAGE
*ENDING
NULL
NUMBER
NUMERIC
NUMERIC-EDITED

OBJECT-COMPUTER
OCCURS
OF
OFF
OMITTED
ON
ONLY
OPEN
OPTIONAL
OR
ORDER
ORGANIZATION
OTHER
*OTHERWISE
OUTPUT
OVERFLOW
OWNER

PADDING
PAGE
PAGE-COUNTER
*PASSWORD
PERFORM
PERMANENT
PF
PH

I-O
I-O-CONTROL
*ID
IDENTIFICATION
IF
IN
INCLUDING
INDEX
INDEX-N
INDEXED
INDICATE
INITIAL
INITIALIZE
INITIATE
INPUT
REALM
REALMS
REALM-NAME
*RECEIVE
RECORD
RECORD-NAME
*RECORD-OVERFLOW
RECORDS
REDEFINES
REEL
REFERENCE-MODIFIER
REFERENCES
RELATIVE
RELEASE
*RELOAD
REMAINDER
REMONITOR
REMOVAL
RENAMES
*REORG-CRITERIA
REPLACING
REPORT
REPORTING
REPORTS
*REREAD
RERUN
RESERVE
RESET
RETAINING
RETRIEVAL
RETURN

MODULES
*MORE-LABELS
MOVE
MULTIPLE
MULTIPLY

*NAMED
NATIVE
NEGATIVE
NEXT
NO
*NOMINAL
NON-NULL
NOT
*NSTD-REELS
*SKIP1
*SKIP2
*SKIP3
SORT
*SORT-CORE-SIZE
*SORT-FILE-SIZE
*SORT-MERGE
*SORT-MESSAGE
*SORT-MODE-SIZE
SORT-OPTION
*SORT-RETURN
SOURCE
SOURCE-COMPUTER
SPACE
SPACES
SPECIAL-NAMES
STANDARD
STANDARD-1
STANDARD-2
START
STATUS
STOP
STORE
STRING
SUB-QUEUE-1
SUB-QUEUE-2
SUB-QUEUE-3
SUB-SCHEMA
SUBTRACT
SUM
SUPPRESS

PIC
PICTURE
PLUS
POINTER
POSITION
•POSITIONING
POSITIVE
•PRINT-SWITCH
PRINTING
PRIOR
PRIVACY
PROCEDURE
PROCEDURES
PROCEED
PROCESS
•PROGRAM
PROGRAM-ID
PROTECTED
PURGE

QUEUE
QUOTE
QUOTES

RANDOM
RD
READ
•READY
TIME
•TIME-OF-DAY
TIMES
TO
TOP
•TOTALED
•TOTALING
•TRACE
•TRACK
•TRACK-AREA
•TRACK-LIMIT
•TRACKS
TRAILING
TRANSFORM
TYPE

UNEQUAL
UNIT

•RETURN-CODE
REVERSED
REWIND
REWRITE
RF
RH
RIGHT
ROUNDED
RUN

SAME
SD
SEARCH
SECTION
SECURITY
SEGMENT
SEGMENT-LIMIT
SELECT
SELECTIVE
SEND
SENTENCE
SEPARATE
SEQUENCE
SEQUENTIAL
•SERVICE
SET
SIGN
SIZE
UNSTRING
UNTIL
UP
UPDATE
UPON
•UPSI-0
•UPSI-1
•UPSI-2
•UPSI-3
•UPSI-4
•UPSI-5
•UPSI-6
•UPSI-7
USAGE
USAGE-MODE
USE
USING

SUSPEND
SYMBOLIC
SYNC
SYNCHRONIZED
•SYSIN
•SYSIPT
•SYSLST
•SYSOUT
•SYSPCH
•SYSPUNCH
•S01
•S02
•S03
•S04
•S05

TABLE
TALLY
TALLYING
TAPE
TENANT
TERMINAL
TERMINATE
TEXT
THAN
•THEN
THROUGH
THRU
VALUE
VALUES
VARYING

WHEN
•WHEN-COMPILED
WITH
WITHIN
WORDS
WORKING-STORAGE
WRITE
•WRITE-ONLY
•WRITE-VERIFY

ZERO
ZEROES
ZEROS

ø Note: This set of reserved words is based on the ANS 74 standard and CODASYL Journal of Development January 1976. IBM extensions are identified with an asterisk.

١٥ - ٣ الصيغ العامة في لغة كوبول *

GENERAL FORMATS

General Format For Identification Division

```
IDENTIFICATION DIVISION.
PROGRAM-ID. program-name.
[AUTHOR. [comment-entry] . . . ]
[INSTALLATION. [comment-entry] . . . ]
[DATE-WRITTEN. [comment-entry] . . . ]
[DATE-COMPILED. [comment-entry] . . . ]
[SECURITY. [comment-entry] . . . ]
```

General Format For Environment Division

```
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. computer-name [WITH DEBUGGING MODE].
OBJECT-COMPUTER. computer-name

  [                       { WORDS      } ]
  [ MEMORY SIZE integer   { CHARACTERS } ]
  [                       { MODULES    } ]

  [PROGRAM COLLATING SEQUENCE IS alphabet-name]
  [SEGMENT-LIMIT IS segment-number].
 [SPECIAL-NAMES. [implementor-name

  { IS mnemonic-name [ON STATUS IS condition-name-1 [OFF STATUS IS condition-name-2]] }
  { IS mnemonic-name [OFF STATUS IS condition-name-2 [ON STATUS IS condition-name-1]] } ] . . .
  { ON STATUS IS condition-name-1 [OFF STATUS IS condition-name-2]                    }
  { OFF STATUS IS condition-name-2 [ON STATUS IS condition-name-1]                    }

  [                    { STANDARD-1                                              } ]
  [                    { NATIVE                                                  } ]
  [ alphabet-name IS   { implementor-name                                        } ] . . .
  [                    { { literal-1 [ { THROUGH } literal-2                ] }  } ]
  [                    { {           [ { THRU    }                          ] }  } ]
  [                    { {           [ ALSO literal-3 [ALSO literal-4] . . .] } . . . } ]
```

---

* المصدر:

Donald A. Sordillo

The Programmer's ANSI Cobol Reference Manual, Prentice - Hall, Inc, New Jersey, 1978,

pp. 365 - 378.

```
    [CURRENCY SIGN IS literal-9]
    [DECIMAL-POINT IS COMMA]•]ᵃ

 ᵃ[INPUT-OUTPUT SECTION•
  FILE-CONTROL•
     {file-control-entry} ...
 ᵇ[I-O-CONTROL•

                                            ⎧⎧[END OF] {REEL} ⎫            ⎫ᶜ
 ᶜ[RERUN[ON{file-name-1        }]  EVERY⎨⎨          {UNIT} ⎬ OF file-name-2⎬ ...
            {implementor-name}           ⎨ integer-1 RECORDS ⎭            ⎬
                                            ⎨integer-2 CLOCK-UNITS          ⎬
                                            ⎩condition-name                 ⎭]

                  [SAME [RECORD    ] AREA FOR {file-name-3} ... ] ...
                        [SORT      ]
                        [SORT-MERGE]

[MULTIPLE FILE TAPE CONTAINS {file-name-5 [POSITION integer-3]} ... ] ... •]ᵇ]ᵃ
```

N.B. The superscripts above the brackets are to indicate matching sets; they are not part of the COBOL language.

General Format for File Control Entry

Format 1

```
SELECT [OPTIONAL] file-name
  ASSIGN TO {implementor-name-1} ...
  [RESERVE integer-1 [AREA ]]
                     [AREAS]
  [ORGANIZATION IS SEQUENTIAL]
  [ACCESS MODE IS SEQUENTIAL]
  [FILE STATUS IS data-name-1]•
```

Format 2

```
SELECT file-name
  ASSIGN TO {implementor-name-1} ...
  [RESERVE integer-1 [AREA ]]
                     [AREAS]
   ORGANIZATION IS RELATIVE

  [ACCESS MODE IS {  SEQUENTIAL [RELATIVE KEY IS data-name-1]}]
                  {{RANDOM }                                 }
                  {{DYNAMIC} RELATIVE KEY IS data-name-1      }

  [FILE STATUS IS data-name-2]•
```

Format 3

```
SELECT file-name
  ASSIGN TO {implementor-name-1} . . .
  [RESERVE integer-1 [AREA
                      AREAS]]
   ORGANIZATION IS INDEXED
  [                  {SEQUENTIAL}]
  [ACCESS MODE IS    {RANDOM    }]
  [                  {DYNAMIC   }]
  RECORD KEY IS data-name-1
 [ALTERNATE RECORD KEY IS data-name-2 [WITH DUPLICATES]] . . .
 [FILE STATUS IS data-name-5].
```

Format 4

```
SELECT file-name ASSIGN TO {implementor-name-1} . . .
```

General Format For Data Division

```
DATA DIVISION.
FILE SECTION.
[FD file-name
  [BLOCK CONTAINS [integer-1 TO] integer-2 {RECORDS   }]
                                           {CHARACTERS}
  [RECORD CONTAINS [integer-3 TO] integer-4 CHARACTERS]
   LABEL {RECORD IS   } {STANDARD}
         {RECORDS ARE } {OMITTED }
  [VALUE OF {implementor-name-1 IS {data-name-1}} . . . ]
                                   {literal-1  }
  [DATA {RECORD IS  } {data-name-3} . . . ]
        {RECORDS ARE}
  [LINAGE IS {data-name-5} LINES [WITH FOOTING AT {data-name-6}]
             {integer-5  }                        {integer-6  }
      [LINES AT TOP {data-name-7}] [LINES AT BOTTOM {data-name-8}]]
                    {integer-7  }                   {integer-8  }
  [CODE-SET IS alphabet-name]
  [{REPORT IS  } {report-name-1} . . . ].
   {REPORTS ARE}
[record-description-entry] . . . ] . . .
[SD file-name
     [RECORD CONTAINS [integer-1 TO] integer-2 CHARACTERS]
     [DATA {RECORD IS  } {data-name-1}].
           {RECORDS ARE}
    {record-description-entry} . . .] . . .
```

```
[WORKING-STORAGE SECTION.
     [77-level-description-entry]
     [record-description-entry  ] ...]
[LINKAGE SECTION.
     [77-level-description-entry]
     [record-description-entry  ] ...]
[COMMUNICATION SECTION.
     [communication-description-entry
     [record-description-entry] ... ] ...]
a[REPORT SECTION.
     b[RD report-name
       [CODE literal-1]
       [{CONTROL IS   } {{data-name-1} ...             }]
       [{CONTROLS ARE } {FINAL [data-name-1] ...       }]
      c[PAGE [LIMIT IS  ] integer-1 [LINE ] [HEADING integer-2]
             [LIMITS ARE]           [LINES]
        [FIRST DETAIL integer-3][LAST DETAIL integer-4]
        [FOOTING integer-5]]c.
     {report-group-description-entry} ... ]b ... ]a
```

N.B. The superscripts above the brackets are to indicate matching sets; they are not part of the COBOL language.

General Format For Data Description Entry

Format 1

```
level-number {data-name-1}
             {FILLER     }
  [REDEFINES data-name-2]
  [{PICTURE} IS character-string]
  [{PIC    }                    ]
  [            {COMPUTATIONAL}]
  [            {COMP         }]
  [[USAGE IS]  {DISPLAY      }]
  [            {INDEX        }]
  [[SIGN IS]{LEADING }[SEPARATE CHARACTER]]
  [         {TRAILING}                   ]
  [OCCURS {integer-1 TO integer-2 TIMES DEPENDING ON data-name-3}
  [       {integer-2 TIMES                                      }
      [{ASCENDING } KEY IS {data-name-4} ...] ...
      [{DESCENDING}                         ]
      [INDEXED BY {index-name-1} ... ]
  [{SYNCHRONIZED}[LEFT ]]
  [{SYNC        }[RIGHT]]
  [{JUSTIFIED} RIGHT]
  [{JUST     }      ]
  [BLANK WHEN ZERO]
  [VALUE IS literal].
```

Format 2

```
66 data-name-1 RENAMES data-name-2 [ {THROUGH} data-name-3 ].
                                     {THRU   }
```

Format 3

```
88 condition-name {VALUE IS   } {literal-1 [ {THROUGH} literal-2 ]} ... .
                  {VALUES ARE }              {THRU   }
```

General Format For Communication Description Entry

Format 1

```
CD cd-name
                      ┌ *[[SYMBOLIC QUEUE IS data-name-1]          ┐
                      │   [SYMBOLIC SUB-QUEUE-1 IS data-name-2]    │
                      │   [SYMBOLIC SUB-QUEUE-2 IS data-name-3]    │
                      │   [SYMBOLIC SUB-QUEUE-3 IS data-name-4]    │
                      │   [MESSAGE DATE IS data-name-5]            │
FOR [INITIAL] INPUT   │   [MESSAGE TIME IS data-name-6]            │
                      │   [SYMBOLIC SOURCE IS data-name-7]         │
                      │   [TEXT LENGTH IS data-name-8]             │
                      │   [END KEY IS data-name-9]                 │
                      │   [STATUS KEY IS data-name-10]             │
                      │   [MESSAGE COUNT IS data-name-11]]*        │
                      └ [data-name-1, data-name-2, ..., data-name-11] ┘.
```

N.B. The superscripts above the brackets are to indicate matching sets; they are not part of the COBOL language.

Format 2

```
CD cd-name FOR OUTPUT
  [DESTINATION COUNT IS data-name-1]
  [TEXT LENGTH IS data-name-2]
  [STATUS KEY IS data-name-3]
  [DESTINATION TABLE OCCURS integer-2 TIMES [INDEXED BY {index-name-1}...]]
  [ERROR KEY IS data-name-4]
  [SYMBOLIC DESTINATION IS data-name-5].
```

General Format For Report Group Description Entry

Format 1

```
01 [data-name-1]
   [ LINE NUMBER IS {integer-1 [ON NEXT PAGE]} ]
                    {PLUS integer-2          }
   [                 {integer-3      } ]
   [ NEXT GROUP IS   {PLUS integer-4 } ]
   [                 {NEXT PAGE      } ]
```

```
            ⎧ {REPORT HEADING}                                  ⎫
            ⎪ {RH            }                                  ⎪
            ⎪ {PAGE HEADING}                                    ⎪
            ⎪ {PH          }                                    ⎪
            ⎪ {CONTROL HEADING} {data-name-2}                   ⎪
            ⎪ {CH             } {FINAL      }                   ⎪
TYPE IS     ⎨ {DETAIL}                                          ⎬
            ⎪ {DE    }                                          ⎪
            ⎪ {CONTROL FOOTING} {data-name-3}                   ⎪
            ⎪ {CF             } {FINAL      }                   ⎪
            ⎪ {PAGE FOOTING}                                    ⎪
            ⎪ {PF          }                                    ⎪
            ⎪ {REPORT FOOTING}                                  ⎪
            ⎩ {RF            }                                  ⎭
[[USAGE IS] DISPLAY].
```

Format 2

```
level-number [data-name-1]
  [LINE NUMBER IS {integer-1 [ON NEXT PAGE]}]
                  {PLUS integer-2          }
  [[USAGE IS] DISPLAY].
```

Format 3

```
level-number [data-name-1]
  [BLANK WHEN ZERO]
  [GROUP INDICATE]
  [{JUSTIFIED} RIGHT]
  [{JUST     }      ]
  [LINE NUMBER IS {integer-1 [ON NEXT PAGE]}]
                  {PLUS integer-2          }
  [COLUMN NUMBER IS integer-3]
   {PICTURE} IS character-string
   {PIC    }
  ⎧ SOURCE IS identifier-1                                       ⎫
  ⎪ VALUE IS literal                                             ⎪
  ⎨ {SUM {identifier-2}... [UPON {data-name-2}...]}...           ⎬
  ⎪ [RESET ON {data-name-4}]                                     ⎪
  ⎩           {FINAL      }                                      ⎭
  [[USAGE IS] DISPLAY].
```

General Format For Procedure Division

Format 1

```
PROCEDURE DIVISION [USING {data-name-1}... ].
[DECLARATIVES.
{section-name SECTION [segment-number]. declarative-sentence
[paragraph-name. [sentence]... ]...}...
END DECLARATIVES.]
```

{section-name SECTION [segment-number].
[paragraph-name. [sentence] . . .] . . .} . . .

Format 2

PROCEDURE DIVISION [USING {data-name-1} . . .].
{paragraph-name. [sentence] . . . } . . .

General Format For Verbs

ACCEPT identifier [FROM mnemonic-name]

ACCEPT identifier FROM {DATE | DAY | TIME}

ACCEPT cd-name MESSAGE COUNT

ADD {identifier-1 | literal-1} . . . TO {identifier-m [ROUNDED]} . . .
[ON SIZE ERROR imperative-statement]

ADD {identifier-1 | literal-1} {identifier-2 | literal-2} . . . GIVING {identifier-m [ROUNDED]} . . .
[ON SIZE ERROR imperative-statement]

ADD {CORRESPONDING | CORR} identifier-1 TO identifier-2 [ROUNDED]
[ON SIZE ERROR imperative-statement]

ALTER {procedure-name-1 TO [PROCEED TO] procedure-name-2} . . .

CALL {identifier-1 | literal-1} [USING {data-name-1} . . .]
[ON OVERFLOW imperative-statement]

CANCEL {identifier-1 | literal-1} . . .

CLOSE {file-name-1 [{REEL | UNIT} [WITH NO REWIND | FOR REMOVAL] | WITH {NO REWIND | LOCK}]} . . .

CLOSE {file-name-1 [WITH LOCK]} . . .

COMPUTE {identifier-1 [ROUNDED]} . . . = arithmetic-expression
[ON SIZE ERROR imperative-statement]

COPY text-name [{OF | IN} library-name]

[REPLACING {{==pseudo-text-1== | identifier-1 | literal-1 | word-1} BY {==pseudo-text-2== | identifier-2 | literal-2 | word-2}} . . .]

DELETE file-name RECORD [INVALID KEY imperative-statement]

DISABLE {INPUT [TERMINAL] | OUTPUT} cd-name WITH KEY {identifier-1 | literal-1}

DISPLAY {identifier-1 | literal-1} . . . [UPON mnemonic-name]

```
DIVIDE {identifier-1} INTO {identifier-2 [ROUNDED]} ...
       {literal-1   }
   [ON SIZE ERROR imperative-statement]

DIVIDE {identifier-1} INTO {identifier-2} GIVING {identifier-3 [ROUNDED]} ...
       {literal-1   }      {literal-2   }
   [ON SIZE ERROR imperative-statement]

DIVIDE {identifier-1} BY {identifier-2} GIVING {identifier-3 [ROUNDED]} ...
       {literal-1   }    {literal-2   }
   [ON SIZE ERROR imperative-statement]

DIVIDE {identifier-1} INTO {identifier-2} GIVING identifier-3 [ROUNDED]
       {literal-1   }      {literal-2   }
   REMAINDER identifier-4 [ON SIZE ERROR imperative-statement]

DIVIDE {identifier-1} BY {identifier-2} GIVING identifier-3 [ROUNDED]
       {literal-1   }    {literal-2   }
   REMAINDER identifier-4 [ON SIZE ERROR imperative-statement]

ENABLE {INPUT [TERMINAL]} cd-name WITH KEY {identifier-1}
       {OUTPUT          }                  {literal-1   }

ENTER language-name [routine-name].
EXIT [PROGRAM].

GENERATE {data-name  }
         {report-name}

GO TO [procedure-name-1]
GO TO {procedure-name-1} ... procedure-name-n
   DEPENDING ON identifier

IF condition {statement-1  } {ELSE statement-2    }
             {NEXT SENTENCE} {ELSE NEXT SENTENCE  }

INITIATE {report-name-1} ...
INSPECT identifier-1 TALLYING

 {identifier-2 FOR {{{ALL    } {identifier-3}} [{BEFORE} INITIAL {identifier-4}]} ...} ...
                   {{{LEADING} {literal-1   }} [{AFTER } INITIAL {literal-2   }]}
                   {{CHARACTERS              }}

INSPECT identifier-1 REPLACING

 {CHARACTERS BY {identifier-6} [{BEFORE} INITIAL {identifier-7}]                        }
 {              {literal-4   } [{AFTER }         {literal-5   }]                        }
 {{ALL    } {{identifier-5} BY {identifier-6} [{BEFORE} INITIAL {identifier-7}]} ...} ...}
 {{LEADING} {{literal-3   }    {literal-4   } [{AFTER }         {literal-5   }]}       }
 {{FIRST  }                                                                            }

INSPECT identifier-1 TALLYING

 {identifier-2 FOR {{{ALL    } {identifier-3}} [{BEFORE} INITIAL {identifier-4}]} ...} ...
                   {{{LEADING} {literal-1   }} [{AFTER }         {literal-2   }]}
                   {{CHARACTERS              }}

 REPLACING

 {CHARACTERS BY {identifier-6} [{BEFORE} INITIAL {identifier-7}]                        }
 {              {literal-4   } [{AFTER }         {literal-5   }]                        }
 {{ALL    } {{identifier-5} BY {identifier-6} [{BEFORE} INITIAL {identifier-7}]} ...} ...}
 {{LEADING} {{literal-3   }    {literal-4   } [{AFTER }         {literal-5   }]}       }
 {{FIRST  }                                                                            }
```

```
MERGE file-name-1 { ON { ASCENDING  } KEY {data-name-1} ... } ...
                       { DESCENDING }
  [COLLATING SEQUENCE IS alphabet-name]
  USING file-name-2 {file-name-3} ...

  { OUTPUT PROCEDURE IS section-name-1 [ { THROUGH } section-name-2 ] }
  {                                      { THRU    }                   }
  { GIVING file-name-5                                                 }

MOVE { identifier-1 } TO {identifier-2} ...
     { literal      }

MOVE { CORRESPONDING } identifier-1 TO identifier-2
     { CORR          }

MULTIPLY { identifier-1 } BY {identifier-2 [ROUNDED]} ...
         { literal-1    }
  [ON SIZE ERROR imperative-statement]

MULTIPLY { identifier-1 } BY { identifier-2 } GIVING {identifier-3 [ROUNDED]} ...
         { literal-1    }    { literal-2    }
  [ON SIZE ERROR imperative-statement]

     { INPUT { file-name-1 [ REVERSED         ] } ...  }
     {                     [ WITH NO REWIND   ]        }
OPEN { OUTPUT {file-name-3 [WITH NO REWIND]} ...       }
     { I-O {file-name-5} ...                           }
     { EXTEND {file-name-7} ...                        }

PERFORM procedure-name-1 [ { THROUGH } procedure-name-2 ]
                           { THRU    }

PERFORM procedure-name-1 [ { THROUGH } procedure-name-2 ] { identifier-1 } TIMES
                           { THRU    }                    { integer-1    }

PERFORM procedure-name-1 [ { THROUGH } procedure-name-2 ] UNTIL condition-1
                           { THRU    }

PERFORM procedure-name-1 [ { THROUGH } procedure-name-2 ]
                           { THRU    }

   VARYING { identifier-2 } FROM { identifier-3 }
           { index-name-1 }      { index-name-2 }
                                 { literal-1    }

     BY { identifier-4 } UNTIL condition-1
        { literal-3    }

[ AFTER { identifier-5 } FROM { identifier-6 }
        { index-name-3 }      { index-name-4 }
                              { literal-3    }

     BY { identifier-7 } UNTIL condition-2 ]
        { literal-4    }

[ AFTER { identifier-8 } FROM { identifier-9 }
        { index-name-5 }      { index-name-6 }
                              { literal-5    }
```

```
BY {identifier-10 | literal-6} UNTIL condition-3 ]

READ file-name [NEXT] RECORD [INTO identifier][AT END imperative-statement]
READ file-name RECORD [INTO identifier] [KEY IS data-name]
    [INVALID KEY imperative-statement]

RECEIVE cd-name {MESSAGE | SEGMENT} INTO identifier-1 [NO DATA imperative-statement]

MULTIPLY {identifier-1 | literal-1} BY {identifier-2 | literal-2} GIVING {identifier-3 [ROUNDED]}...
    [ON SIZE ERROR imperative-statement]

     { INPUT {file-name-1 [REVERSED | WITH NO REWIND]}...   }
OPEN { OUTPUT {file-name-3 [WITH NO REWIND]}...             }
     { I-O {file-name-5}...                                 }
     { EXTEND {file-name-7}...                              }

PERFORM procedure-name-1 [{THROUGH | THRU} procedure-name-2]

PERFORM procedure-name-1 [{THROUGH | THRU} procedure-name-2] {identifier-1 | integer-1} TIMES

PERFORM procedure-name-1 [{THROUGH | THRU} procedure-name-2] UNTIL condition-1

PERFORM procedure-name-1 [{THROUGH | THRU} procedure-name-2]
    VARYING {identifier-2 | index-name-1} FROM {identifier-3 | index-name-2 | literal-1}
      BY {identifier-4 | literal-3} UNTIL condition-1
[ AFTER {identifier-5 | index-name-3} FROM {identifier-6 | index-name-4 | literal-3}
      BY {identifier-7 | literal-4} UNTIL condition-2 ]
[ AFTER {identifier-8 | index-name-5} FROM {identifier-9 | index-name-6 | literal-5}
```

```
      BY {identifier-10 | literal-6} UNTIL condition-3 ]

READ file-name [NEXT] RECORD [INTO identifier][AT END imperative-statement]
READ file-name RECORD [INTO identifier] [KEY IS data-name]
    [INVALID KEY imperative-statement]
```

```
RECEIVE cd-name {MESSAGE} INTO identifier-1 [NO DATA imperative-statement]
                {SEGMENT}

RELEASE record-name [FROM identifier]
RETURN file-name RECORD [INTO identifier] AT END imperative-statement
REWRITE record-name [FROM identifier][INVALID KEY imperative-statement]

SEARCH identifier-1 [VARYING {identifier-2  }][AT END imperative-statement-1]
                             {index-name-1  }

    {WHEN condition-1 {imperative-statement-2}} ...
                      {NEXT SENTENCE         }

SEARCH ALL identifier-1 [AT END imperative-statement-1]

     WHEN {data-name-1 {IS EQUAL TO} {identifier-3             }}
          {            {IS =       } {literal-1                }}
          {                          {arithmetic-expression-1  }}
          {condition-name-1                                     }

          [AND {data-name-2 {IS EQUAL TO} {identifier-4            }}] ...
          [    {            {IS =       } {literal-2               }}]
          [    {                          {arithmetic-expression-2 }}]
          [    {condition-name-2                                    }]

          {imperative-statement-2}
          {NEXT SENTENCE         }

SEND cd-name FROM identifier-1

                                  {WITH identifier-2}
SEND cd-name [FROM identifier-1]  {WITH ESI         }
                                  {WITH EMI         }
                                  {WITH EGI         }

  [{BEFORE} ADVANCING {{identifier-3} [LINE ]}]
  [{AFTER }           {{integer     } [LINES]}]
  [                   {{mnemonic-name}       }]
  [                   {{PAGE         }       }]

                                      {identifier-3  }
SET {identifier-1  } ... TO           {index-name-3  }
    {index-name-1  }                  {integer-1     }

SET {index-name-4} ... {UP BY  } {identifier-4}
                       {DOWN BY} {integer-2   }

SORT file-name-1 {ON {ASCENDING } KEY {data-name-1} ...} ...
                     {DESCENDING}
  [COLLATING SEQUENCE IS alphabet-name]

  {INPUT PROCEDURE IS section-name-1 [{THROUGH} section-name-2]}
  {                                   {THRU   }                }
  {USING {file-name-2} ...                                     }

  {OUTPUT PROCEDURE IS section-name-3 [{THROUGH} section-name-4]}
  {                                    {THRU   }                }
  {GIVING file-name-4                                           }
```

```
                              ⎡     ⎧ IS EQUAL TO          ⎫          ⎤
                              ⎢     ⎪ IS =                 ⎪          ⎥
START file-name               ⎢ KEY ⎨ IS GREATER THAN      ⎬ data-name⎥
                              ⎢     ⎪ IS >                 ⎪          ⎥
                              ⎢     ⎪ IS NOT LESS THAN     ⎪          ⎥
                              ⎣     ⎩ IS NOT <             ⎭          ⎦
  [INVALID KEY imperative-statement]

STOP {RUN    }
     {literal}

STRING {{identifier-1}}...DELIMITED BY {identifier-3}}...
       {{literal-1   }}                {literal-3   }
                                       {SIZE        }
   INTO identifier-7 [WITH POINTER identifier-8]
   [ON OVERFLOW imperative-statement]

SUBTRACT {identifier-1}...FROM {identifier-m [ROUNDED]}...
         {literal-1   }
   [ON SIZE ERROR imperative-statement]

SUBTRACT {identifier-1}...FROM {identifier-m}
         {literal-1   }        {literal-m   }
   GIVING {identifier-n [ROUNDED]}...
   [ON SIZE ERROR imperative-statement]

SUBTRACT {CORRESPONDING} identifier-1 FROM identifier-2 [ROUNDED]
         {CORR         }
   [ON SIZE ERROR imperative-statement]

SUPPRESS PRINTING

TERMINATE {report-name-1}...

UNSTRING identifier-1
   [DELIMITED BY [ALL]{identifier-2}[OR [ALL]{identifier-3}]...]
                      {literal-1   }          {literal-2   }
     INTO {identifier-4 [DELIMITER IN identifier-5][COUNT IN identifier-6]}...
   [WITH POINTER identifier-10][TALLYING IN identifier-11]
   [ON OVERFLOW imperative-statement]

                                                             ⎧ {file-name-1}... ⎫
                                                             ⎪ INPUT            ⎪
USE AFTER STANDARD {EXCEPTION} PROCEDURE ON                  ⎨ OUTPUT           ⎬
                   {ERROR    }                               ⎪ I-O              ⎪
                                                             ⎩ EXTEND           ⎭.

USE BEFORE REPORTING identifier.

                       ⎧ cd-name-1                             ⎫
                       ⎪ [ALL REFERENCES OF] identifier-1      ⎪
USE FOR DEBUGGING ON   ⎨ file-name-1                           ⎬
                       ⎪ procedure-name-1                      ⎪
                       ⎩ ALL PROCEDURES                        ⎭....

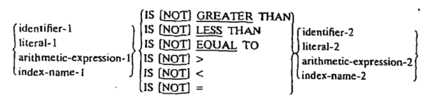
WRITE record-name [FROM identifier-1]
   ⎡                         ⎧ {identifier-2}{LINE } ⎫ ⎤
   ⎢ {BEFORE}                ⎪ {integer     }{LINES} ⎪ ⎥
   ⎢ {AFTER } ADVANCING      ⎨                       ⎬ ⎥
   ⎢                         ⎪ {mnemonic-name}       ⎪ ⎥
   ⎣                         ⎩ {PAGE         }       ⎭ ⎦
```

[AT {END-OF-PAGE | EOP} imperative-statement]

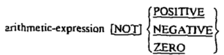
WRITE record-name [FROM identifier][INVALID KEY imperative-statement]

General Format For Conditions

Relation Condition


{identifier-1 | literal-1 | arithmetic-expression-1 | index-name-1} {IS [NOT] GREATER THAN | IS [NOT] LESS THAN | IS [NOT] EQUAL TO | IS [NOT] > | IS [NOT] < | IS [NOT] =} {identifier-2 | literal-2 | arithmetic-expression-2 | index-name-2}

Class condition


identifier IS [NOT] {NUMERIC | ALPHABETIC}

Sign condition

arithmetic-expression [NOT] {POSITIVE | NEGATIVE | ZERO}

Condition-name condition

condition-name

Switch-status condition

condition-name

Negated simple condition

NOT simple-condition

Combined condition

condition {{AND | OR} condition}...

Abbreviated combined relation condition

relation-condition {{AND | OR} [NOT] [relational-operator] object}...

Miscellaneous Formats

Qualification

{data-name-1 | condition-name} [{OF | IN} data-name-2]...

paragraph-name [{OF | IN} section-name]

text-name [{OF | IN} library-name]

Subscripting

{data-name | condition-name} (subscript-1 [, subscript-2 [, subscript-3]])

```
Indexing
        {data-name      }({ index-name-1 [{±} literal-2] }
        {condition-name } ({ literal-1                   }
            [ , { index-name-2 [{±} literal-4] }][ , { index-name-3 [{±} literal-6] }])
            [   { literal-3                    }][   { literal-5                    }])

Identifier Format 1
            data-name-1 [{OF} data-name-2] ... [(subscript-1 [ , subscript-2
                         {IN}
                [ , subscript-3]])]

Identifier Format 2
        data-name-1 [{OF} data-name-2] ... [({ index-name-1 [{±} literal-2] }
                     {IN}                    { literal-1                    }
            [ , { index-name-2 [{±} literal-4] }][ , { index-name-3 [{±} literal-6] }])]
            [   { literal-3                    }][   { literal-5                    }])]
```

١٥ ـ ٤ معجم لأهم المصطلحات* المستخدمة في لغة كوبول :

طريقة المعالجة ACCESS MODE

وهي الطريقة التي تعالج بها السجلات ضمن الملف

الفاصلة العشرية الفعلية ACTUAL DECIMAL POINT

التمثيل الفعلي للفاصلة العشرية في الحقل وقد تكون اشارة (.) أو (,) .

الرمز الهجائي ALPHABETIC CHARACTERS

وهو الرمز الذي يدل على أحد الحروف في المجموعة التالية :

A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q,

R, S, T, U, V, W, X, Y, Z, SPACE والفراغ

الرمز الهجائي / الرقمي المختلط ALPHANUMERIC CHARACTER

عبارة عن أي رمز يستخدم في مجموعة رموز الحاسب .

* Introductory Structured Cobol Programming, By Gray S. Popkin.

تعبير رياضي (حسابي) ARITHMATIC EXPRESSION

يمكن أن يكون التعبير الحسابي حقلاً رقمياً أو ثابتاً رقمياً أو مجموعة من كليهما تفصل بينهما إشارة عملية حسابية أو يحيط بهما أقواس.

عامل حسابي (رياضي) ARITHMATIC OPERATOR

هو رمز (أو رمزان مندمجان) يدل على ما يلي :

الرمــز	المعنــى
+	إضافة
—	طرح
*	ضرب
/	قسمة
**	الرفع إلى قوّة (أس).

المفتاح التصاعدي ASCENDING KEY

وهو حقل يحتوي على قيمة ترتب وفقها البيانات بصورة تصاعدية، تبدأ بالقيمة الأقل وتنتهي بالقيمة الأعلى.

الفاصلة العشرية الافتراضية ASSUMED DECIMAL POINT

وهي المكان الذي يفترض أن توجد فيه الفاصلة العشرية إلاّ أنها لا تمثل برمز فعلياً.

شرط النهاية AT END CONDITION

يكتب هذا الشرط في الحالات التالية :

١ ــ أثناء تنفيذ تعليمة القراءة READ لملف معالج بالطريقة التسلسلية.

٢ ــ أثناء تنفيذ تعليمة العودة RETURN عندما لا يوجد سجل تالي للمعالجة

بالنسبة للملفات الدمج والفرز SORT AND MERGE

٣ ــ أثناء تنفيذ تعليمة البحث SEARCH عندما تنتهي عمليات البحث دون وجود القيمة المحددة بعبارة WHEN

كتلة بيانات BLOCK

وهي مجموعة من السجلات، وكل مجموعة تتألف من سجل واحد أو أكثر.

رمــز CHARACTER

وهو أصغر وحدة أساسية من اللغة.

سلسلة الرموز CHARACTER STRING

وهي عبارة عن سلسلة متصلة من الرموز تشكل إما كلمة كوبول أو ثابتاً حرفياً، أو شكل PICTURE أو ملاحظة.

شرط النوع CLASS CONDITION

وهو الافتراض الذي من أجله يمكن للقيمة الحقيقية أن تقرر بأن قيمة حقل معين بكامله إما ابجدياً أو رقمياً.

مجموعة رموز لغة كوبول COBOL CHARACTER SET

إن مجموعة الرموز المستخدمة في لغة كوبول هي ٥١ (واحد وخمسين) رمزاً مصنفة كالتالي:

الرمــز	المعنــى
الأرقام : من الصفر إلى 9	الأرقام من الصفر إلى التسعة
A → Z	الأحرف الهجائية من A إلى Z

الفراغ (ويمثل بحرف واحد)	
إشارة الجمع	+
إشارة الناقص (أو الشرطة)	—
إشارة النجمة	*
إشارة الخط المائل	/
إشارة المساواة	=
إشارة الدولار	$
الفاصلة	,
الفاصلة المنقوطة	؛
النقطة (الفاصلة العشرية)	.
علامة الاقتباس	"
القوس الأيسر	(
القوس الأيمن	)
إشارة أكبر من	>
إشارة أصغر من	<

عمود (موقع) COLUMN

هو موقع الرمز في سطر الطباعة، و يرقم ابتداءاً من الرقم ١ أقصى يسار السطر ولغاية الجهة اليمنى نهاية السطر.

الشرط المركب COMBINED CONDITION

وهو حاصل ربط شرطين أو أكثر بعملية AND أو OR المنطقية.

سطر الملاحظات COMMENT LINE

يوجد هذا السطر في برنامج كوبول مكتوبا بصيغة المصدر (الأصل) SOURCE و يعرّف بوضع علامة * في العمود رقم سبعة.

و يوضع هذا السطر من أجل شرح العمليات في البرنامج بصورة موجزة ويخدم في عملية توثيق البرنامج.

اسم الشرط CONDITION NAME

وهو اسم لحقل يحدد فيه المبرمج قيمة معينة أو مجموعة قيم أو مدى من القيم يمكن للمتغير الشرطي أن يأخذها.

التعبير الشرطي CONDITIONAL EXPRESSION

وهو شرط بسيط أو شرط مركب يعين بتعليمة IF أو PERFORM أو SEARCH

فصل مكونات الحاسب CONFIGURATION SECTION

وهو جزء من قسم المحيطات ENVIRONMENT DIVISION الذي يبين مواصفات الحاسب المستخدم.

الحقول المتتابعة CONTIGUOUS ITEMS (FIELDS)

وهي الحقول التي توصف بشكل مداخل متعاقبة في قسم البيانات DATA DIVISION وتظهر ذات علاقة هرمية مع بعضها البعض.

CONTROL BREAK

وهو التغير الذي يحصل على قيمة حقل معتبر كحقل تحكم في عبارة CONTROL

العداد COUNTER

عبارة عن حقل يستخدم لتخزين الأرقام التي تمثل بطريقة تسمح لهذه الأرقام بالزيادة أو النقصان بوساطة قيمة أخرى. أو تبديل هذه الأرقام أو إعادتها للقيمة صفر أو لأي قيمة اعتباطية موجبة أو سالبة.

السجل الحالي CURRENT RECORD

وهو السجل الذي يوجد في المنطقة المخصصة للملف في الذاكرة من أجل إجراء العمليات اللازمة عليه.

عنصر البيانات DATA ITEM

وهو رمز أو مجموعة رموز متصلة يمكن تعريفها على أنها وحدة من البيانات في برنامج كوبول.

اسم البيان (حقل) DATA NAME

عبارة عن كلمة تحدد من قبل المبرمج لتسمية عنصر البيانات الموصوف في قسم البيانات DATA DIVISION

رمز النهاية DELIMITER

رمز أو سلسلة رموز متصلة التي يمكن أن تحدد نهاية سلسلة من الرمز وتفصلها عن سلسلة أخرى تالية.

المفتاح التنازلي DESCENDING KEY

وهو حقل أو مفتاح يحتوي على قيمة ترتب وفقها البيانات بصورة تنازلية تبدأ

بالقيمة الأعلى وتنتهي بالقيمة الأقل.

قسم (من برنامج كوبول) DIVISION

يتكون من فصل واحد أو أكثر وكل فصل يتكون من مجموعة فقرات يدعى هيكل القسم ويمكن الاّ يحتوي القسم على أي فصل.

وهناك ٤ أقسام في برنامج كوبول وهي :

قسم التعريف IDENTIFICATION DIVISION

قسم المحيطات ENVIRONMENT DIVISION

قسم البيانات DATA DIVISION

قسم الاجراءات PROCEDURE DIVISION

رمز الاخراج EDITING CHARACTER

وهو عبارة عن رمز أو رمزين ثابتين يشكلان ما يلي :

الرمــز	المعنــى
B	فــراغ
∅	صــفر
+	الاشارة الجبرية الموجبة
—	الاشارة الجبرية السالبة
CR	إشارة دائن
DB	إشارة مدين
Z	طمس الصفر (تمثيل الصفر على يسار الرقم بفراغ)
$	إشارة الدولار
,	الفاصلة (الفاصلة العشرية)

النقطة (الفاصلة العشرية)

الخط المائل

العنصر الأولي ELEMENTARY ITEM

وهو عنصر البيانات الموصوف بشكل لا يمكن أن يجزأ من الناحية النظرية.

الجملة المستقلة ENTRY

عبارة عن مجموعة متعاقبة من العبارات الوصفية تنتهي بنقطة وتوجد في أقسام التعريف والمحيطات والبيانات في برنامج كوبول.

الثابت الشكلي FIGURATIVE CONSTANT

وهي قيم تولد عن المترجم COMPILER وتعتبر ككلمات محجوزة (مثل SPACE وZERO ...).

الملف FILE

وهو مجموعة من السجلات.

تنظيم الملف FILE ORGANIZATION

وهي البيئة النظرية الثابتة التي أعطيت للملف عند إنشائه.

فصل توصيف الملفات FILE SECTION

عبارة عن فصل في قسم البيانات ويحتوي على جمل وصفية وللملف مع توصيفات السجل الخاصة بالملف.

شكل، صيغة FORMAT

وهي ترتيب محدد لمجموعة من البيانات.

مجموع عناصر بيانات GROUP ITEM

وهي مجموعة متصلة من عناصر أولية أو من عناصر مجمعة للبيانات وتعطى اسماً واحداً.

النهاية الأعظمية HIGH ORDER END

وهو الحرف الموجود في أقصى يسار سلسلة رموز.

طريقة إدخال ـ إخراج I-O MODE

وهي حالة الملف بعد تنفيذ تعليمة فتح الملف OPEN بعبارة I-O (إدخال ـ إخراج) وقبل إغلاق الملف بتعليمة CLOSE

اسم حقل (بيان) IDENTIFIER

اسم لبيان يتبع لقواعد لغة كوبول من حيث تشكيله، ويمكن أن يكون مركباً من أسماء معرفة وأدلة ومؤشرات لازمة لجعله مرجعاً وحيداً لعنصر البيانات في برنامج كوبول.

التعليمة الأمرية IMPERATIVE STATEMENT

وهي التعليمة التي تبدأ بفعل أمر وتعين عملاً غير مشروط يجب أن ينفذ ويمكن للتعليمة الآمرة أن تحتوي على سلسلة من التعليمات الآمرة الأخرى.

الفهرس (المؤشر) INDEX

وهو موقع محدد في ذاكرة الحاسب، وتمثل محتو ياته تعريف لعنصر محدد في جدول TABLE

اسم الفهرس (المؤشر) INDEX-NAME

كلمة تعرف من قبل المبرمج لتسمية مؤشر أو فهرس لجدول محدد.

الملف المفهرس INDEX FILE

وهو الملف ذو التنظيم المفهرس.

التنظيم المفهرس INDEXED ORGANIZATION

وهو الهيكل النظري الثابت للملف، حيث يعرف كل سجل فيه بقيمة لمفتاح واحد أو أكثر ضمن سجل هذا الملف.

ملف الادخال INPUT FILE

هو الملف المفتوح للمعالجة بطريقة الادخال (و يستخدم للقراءة فقط).

صيغة الادخال INPUT MODE

وهي حالة الملف بعد تنفيذ تعليمة OPEN بعبارة INPUT وقبل تنفيذ تعليمة إغلاق الملف CLOSE

ملف إدخال ـ إخراج INPUT-OUTPUT FILE

وهو الملف المفتوح للمعالجة بصيغة إدخال ـ إخراج (ويمكن القراءة من هذا الملف

والكتابة عليه).

فصل الادخال ـ الاخراج INPUT-OUTPUT SECTION

وهو فصل من قسم المحيطات ENVIRONMENT DIVISION في برنامج كوبول ويستخدم لتسمية الملفات والوساطات الخارجية اللازمة لاستخدامات البرنامج والتي تزود البرنامج بالمعلومات المطلوبة للمعالجة أثناء تنفيذ البرنامج.

العدد الصحيح INTEGER

وهو الحقل الرقمي أو العدد الذي لا يحتوي في الجزء الأيمن من الفاصلة العشرية المفترضة على أي رقم.

شرط عدم صلاحية المفتاح INVALID KEY CONDITION

يتحقق هذا الشرط عندما تكون القيمة المحددة للمفتاح غير مطابقة لمحتويات الفهرس في الملفات النسبية والمفهرسة.

المفتاح KEY

وهو الحقل الذي يعرّف موقعاً من السجل، أو هو مجموعة الحقول التي تحدد تسلسل البيانات.

رقم المستوى LEVEL NUMBER

وهو رقم محدد من قبل المبرمج و يشير إلى موقع عنصر البيانات من البنية الهرمية للسجل، أو هو الرقم الذي يشير إلى مميزات خاصة لهذا العنصر. ويمثل رقم المستوى بخانة واحدة أو خانتين رقميتين وتتراوح قيمتها من ١ إلى ٤٩ مشيراً بذلك إلى البنية

الهرمية لعنصر السجل. والأرقام من ١ ــ ٩ قد تمثل مفردة أو قد يسبقها صفر.

حرفي (قيمة ثابتة) LITERAL

هي سلسلة الرموز التي تكون قيمتها هي نفس الرموز المكونة لها ولا تحتاج إلى اسم متغير.

العامل المنطقي LOGICAL OPERATOR

هو أحد الكلمات المحجوزة التالية AND أو OR أو NOT المصاغ في شكل شرط، إن كلاً من أو احدى الكلمتين OR و AND تستخدم بعمليات الربط المنطقية، أما كلمة NOT فتستخدم من أجل النفي المنطقي.

السجل المنطقي LOGICAL RECORD

وهو الاسم الشامل لمجموعة عناصر بيانات و يشار إليه برقم مستوى 01 دوماً.

النهاية الدنيا LOW ORDER END

وهي الحرف الموجود في أقصى يمين سلسلة من الرموز.

تخزين كتل البيانات MASS STORAGE

وهو وسط التخزين الذي يمكن للبيانات أن تنظم أو تعالج عليه بطريقة تسلسلية وغير تسلسلية.

ملف تخزين كتل البيانات MASS STORAGE FILE

وهو مجموعة من السجلات موجودة على وسط تخزين خاص بكتل البيانات.

الاسم الرمزي (المنطقي) لوسط تخزين MNEMONIC NAME

هي الكلمة التي يحددها المبرمج في قسم المحيطات في برنامج كوبول وتشير إلى وسط تخزين محدد.

الشرط المركب المنفي NEGATED COMBINED CONDITION

وهو عامل النفي NOT متبعاً بشرط مركب محصور ضمن قوسين.

الشرط البسيط المنفي NEGATED SIMPLE CONDITION

وهو عامل النفي NOT متبعاً بشرط بسيط.

الجملة المنفذة التالية NEXT EXECUTABLE SENTENCE

وهي الجملة التالية التي سينتقل إليها تسلسل تنفيذ البرنامج بعد أن ينتهي من تنفيذ الجملة الحالية.

عناصر البيانات المستقلة NON CONTIGUOUS ITEMS

وهي عناصر أولية للبيانات، غير قابلة للتجزئة موصفة في فصل التخزين الداخلي WORKING-STORAGE SECTION والتي ليس لها علاقات هرمية مع عناصر أخرى، وتأخذ رقم المستوى 77 أو 01.

عنصر البيانات غير الرقمي NON NUMERIC ITEM

وهو العنصر الموصوف بشكل يسمح بأن يحتوي على أي تركيبة من الرموز المصنفه ضمن مجموعة الرموز المستخدمة في الحاسب الآلي.

السلسلة غير الرقمية (الثابت غير الرقمي) NON NUMERIC LITERAL

وهـي سـلسـلـة مـن الـرموز محددة بعلامتي اقتباس" " ويمكن أن تحتوي سلسلة الرموز هذه على أي رمز من مجموعة الرموز المستخدمة في الحاسب الآلي.

الرمز الرقمي NUMERIC CHARACTER

وهي أي رمز يخص مجموعة الرموز التالية :

ϕ , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9

عنصر البيانات الرقمي NUMERIC ITEM

وهو عنصر البيانات الموصف بشكل يقتصر محتواه على قيم رقمية فقط أي من الصفر إلى الـتـسـعـة ، وإذا كـان مـوصـفـاً باشارة فيمكن أن يحتوي على إشارة جبرية موجبة أو سالبة.

رموز رقمية (ثابت رقمي) NUMERIC LITERAL

وهي سلسلة رموز تتألف من رقم واحد أو أكثر أو يمكن أن تكون السلسلة عشرية أو تحـتـوي على إشـارة جـبـريـة مـوجـبـة أو سالبة أو قد تحتوي على فاصلة عشرية مع إشارة جـبـريـة ، والفاصلة العشرية يجب الاّ تكون أول حرف من اليمين (بل ضمن السلسلة) أما الاشارة الجبرية إن وجدت فانها يجب أن تكون بأقصى يسار السلسلة.

البرنامج بلغة الآلة OBJECT PROGRAM

وهـو مجموعة من أوامر التنفيذ الخاصة بلغة الآلة مع أشياء أخرى مصممة لأن تكون الـوسيط في معالجة البيانات ، وتشير عبارة OBJECT PROGRAM بصورة عامة إلى ناتج عملية الترجمة COMPILE لبرنامج مكتوب بلغة كوبول.

صيغة فتح ملف OPEN MODE

وهي حالة الملف بعد تنفيذ تعليمة فتح الملف OPEN وقبل تنفيذ تعليمة إغلاق الملف CLOSE .

وتحدد صيغة الفتح مع تعليمة OPEN بأحد الأشكال التالية :

يستخدم عندها الملف كادخال INPUT

يستخدم عندها الملف كاخراج OUTPUT

يستخدم الملف كادخال ــ إخراج I-O

يستخدم الملف كاخراج EXTEND

عامل البيانات OPERAND

إن التعريف العام لعبارة عامل البيانات هي أنها أي عنصر يتم العمل عليه .

ومن أجل أغراض الحاسب الآلي تعتبر أي كلمة ترد في التعليمة تمثل اسماً لمخزن البيانات هي عامل البيانات OPERAND

الاشارة الجبرية (العملية) OPERATIONAL SIGN

وهي إشارة جبرية ملحقة مع حقل رقمي أو سلسلة رقمية لتدل على هذه القيمة إما موجبة أو سالبة .

ملف إخراج OUTPUT FILE

وهو الملف المفتوح بصيغة الاخراج (للكتابة عليه) .

صيغة الاخراج OUTPUT MODE

وهي حالة الملف بعد تنفيذ تعليمة OPEN بكلمة OUTPUT وقبل تنفيذ تعليمة

إغلاق الملف CLOSE

صفحة PAGE

وهي القسم العمودي من التقرير الذي يمثل البيانات والحقول المفصولة عملياً بفراغات مناسبة.

جسم الصفحة PAGE BODY

وهو الجزء النظري من الصفحة الذي تكتب فيه الأسطر.

أسفل الصفحة PAGE FOOTING

وهو الجزء السفلي من الصفحة والذي يمثل نهاية الصفحة و يرد في نظام منشيء التقارير.

رأس الصفحة PAGE HEADING

وهو الجزء العلوي الذي يمثل بداية الصفحة ويحدد بنظام منشيء التقارير.

الفقرة PARAGRAPH

تمثل الفقرة مجموعة تعليمات في قسم الاجراءات تنتهي بنقطة يليها فراغ ويمكن أن يخصص لها أسم وهذا الأسم ينتهي بنقطة أيضاً أما في قسم التعريف أو المحيطيات فتمثل الفقرة جملة مستقلة واحدة أو أكثر وتسبق باسم لها وتنتهي بنقطة.

مفتاح سجل رئيسي PRIME RECORD KEY

وهو المفتاح الذي يحتوي على قيمة وحيدة تُعرّف السجل ضمن الملف الرئيسي.

اسم الاجراء PROCEDURE NAME

وهو الاسم الذي يختاره المبرمج ليكون اسماً للفقرة، أو للفصل في قسم الاجراءات، ويمكن لهذا الاسم أن يكون معرّفاً.

اسم برنامج PROGRAM NAME

وهي الكلمة التي يختارها المبرمج لتكون اسماً لبرنامجه المكتوب بلغة كوبول.

رموز التنقيط PUNCTUATION CHARACTERS

وهـــي :

الرمــز	المعنـــى
,	الفاصلة
;	الفاصلة المنقوطة
.	النقطة
"	علامة الاقتباس
(	قوس أيسر
)	قوس أيمن
	فــراغ
=	علامة المساواة

اسم عنصر البيانات المكرر QUALIFIED DATA-NAME

وهو اسم عنصر البيانات المؤلف من اسم حقل يليه أحد كلمتي OF أو IN للدلالة على تبعيته لحقل إجمالي آخر (المرجع).

اسم المرجع (المتحول) QUALIFIER

١ ــ هو اسم حقل يستخدم كمرجع لاسم حقل آخر مكرر.

٢ ــ هو اسم فصل SECTION يستخدم كمرجع لاسم فقرة موجودة ضمن هذا الفصل.

طريقة الوصول العشوائية RANDOM ACCESS

وهي الطريقة التي يحدد فيها المبرمج قيمة مفتاح السجل الذي سيعالج في ملف نسبي أو مفهرس ضمن البرنامج.

السجل RECORD

ارجع إلى كلمة LOGICAL RECORD

منطقة السجل RECORD AREA

وهي منطقة مخصصة في الذاكرة الداخلية للحاسب، وذلك لمعالجة السجل الموصف في FD في قسم الملفات FILE SECTION

جملة توصيف ملف RECORD DESCRIPTION ENTRY

هي مجموعة عبارات لتوصيف بيانات سجل معين.

مفتاح السجل RECORD KEY

وهو الحقل الذي بقيمته يعرف السجل في الملف المفهرس.

اسم السجل RECORD NAME

وهي الكلمة التي يختارها المبرمج لتكون اسماً للسجل الموصف في FILE DESCRIPTION (FD).

رموز تحديد العلاقة RELATIVE CONDITION

وهو الافتراض الذي تحدد فيه القيمة الحقيقية بأن قيمة التعبير الرياضي أو عنصر البيانات له علاقة محددة مع قيمة لتعبير رياضي آخر أو لعنصر بيانات آخر.

عامل العلاقة RELATIONAL OPERATION

وهي كلمة محجوزة أو رمز تحديد علاقة أو مجموعة من الكلمات المحجوزة المتتابعة أو مجموعة من رموز تحديد العلاقة والكلمات المحجوزة المتتابعة معاً تستخدم كوصف للشرط وهي :

الرمز	المعنى
IS [NOT] GREATER THAN	أكبر من أو
IS [NOT] >	ليس أكبر من
IS [NOT] LESS THAN	أصغر من أو
IS [NOT] < THAN	ليس أصغر من
IS [NOT] EQUAL TO	مساوٍ أو
IS [NOT] =	غير مساوٍ

الملف النسبي RELATIVE FILE

هو الملف ذو التنظيم النسبي.

التنظيم النسبي RELATIVE ORGANIZATION

وهو الملف المبني أساساً بحيث أن كل سجل فيه معرّف بصورة وحيدة برقم صحيح أكبر من الصفر و يدل على تسلسل هذا السجل في الملف.

جملة توصيف تقرير REPORT DESCRIPTION ENTRY

وهي جملة مستقلة في فصل التقرير REPORT SECTION من قسم البيانات DATA DIVISION مؤلفة من مؤشر مستوى هو RD يليه اسم التقرير يليه مجموعة من عبارات التقرير.

ملف تقرير REPORT FILE

وهو ملف إخراج والذي توصيفه في قسم البيانات يحتوي على عبارة REPORT ومحتوى ملف التقرير هذا هو سجلات مكتوبة باشراف نظام منشيء التقارير REPORT WRITTER CONTROL SYSTEM

الكلمة المحجوزة RESERVED WORD

وهي كلمة في لغة كوبول مدرجة ضمن جدول للكلمات تستخدم في برنامج كوبول ولكن يجب ألاّ تظهر في البرنامج كأسم مستخدم من قبل المبرمج.

فصل SECTION

وهو مجموعة من الفقرات أو الجمل المستقلة يسبقها اسم يعتبر عنواناً لها تدعى فصل ويمكن أن يرد عنوان الفصل دون أن يشمل على فقرات أو جمل.

جملة SENTENCE

وهي عبارة عن تعليمة واحدة أو أكثر تنتهي بنقطة و يليها فراغ واحد على الأقل.

فاصــل SEPARATOR

وهو أحد رموز التنقيط و يستخدم للدلالة على انتهاء سلسلة من الرموز.

طريقة الوصول التسلسلية SEQUENTIAL ACCESS

وهـي طـريـقـة المعـالجة الـتـي يـتـم فـيـها استدعاء السجلات من الملف أو وضع السجلات ضمن الملف بشكل متعاقب حسب تسلسلها في الملف.

الملف المتسلسل SEQUENTIAL FILE

وهو الملف المنظم بطريقة تسلسلية.

تنظيم تسلسلي SEQUENTIAL ORGANIZATION

هـي بُـنْـيَة الملف التي تعتمد بأن كل سجل فيه يعرف بعلاقته إلى سابقه وتاليه من السجلات الموجودة في الملف.

شرط الاشارة SIGN CONDITION

وهـو الافـتـراض الـذي تقرر فيه القيمة الجبرية للحقل الرقمي أو للتعبير الرياضي، وهل هو أصغر أو أكبر أو مساوٍ للصفر.

ملف فرز SORT FILE

وهـو مجـمـوعـة مـن الـسـجـلات المراد فرزها بتعليمة فرزSORT و ينشأ ملف الفرز بـاسـتـخـدام تـعـلـيـمـات الـفـرز فقط، و يعتبر كمنطقة عمل أو تخزين مؤقتة للبيانات المطلوب فرزها.

جملة توصيف ملف فرز دمج SORT MERGE FILE DESCRIPTION ENTRY

وهـي جملة ترد في فصل الملفات FILE SECTION من قسم البيانات تبدأ بمؤشر هو SD و يتبعها اسم الملف و يليها عبارات توصيف الملف .

برنامج مكتوب بصيغة المصدر SOURCE PROGRAM

وهـو الـبـرنـامـج المكـتـوب بلغة المصدر ويمكن أن يحتوي على رموز وأشكال خاصة ، لكنها تتبع قواعد لغة كوبول .

رموز خاصة SPECIAL CHARACTER

وهي الرموز التالية :

الرمــز	المعنــى
+	إشارة الاضافة
—	إشارة الطرح
*	نجمة (إشارة ضرب)
/	إشارة الخط المائل (إشارة القسمة)
=	إشارة المساواة
$	إشارة العملة (الدولار)
,	الفاصلة
;	الفاصلة المنقوطة
.	النقطة (الفاصلة العشرية)
"	علامة الاقتباس
(	قوس أيسر
)	قوس أيمن

< إشارة أكبر من

> إشارة أصغر من

أسماء خاصة SPECIAL NAMES

وهي اسم فقرة في قسم المحيطات ENVIRONMENT DIVISION والتي تشير إلى أسماء معطاة من قبل المبرمج بدلاً من الأسماء الرمزية (المنطقية) المعرُفة سابقاً.

تعليمة STATEMENT

مجموعة من الكلمات والاشارات المكتوبة بصيغة قاعدية صحيحة في قسم الاجراءات وتبدأ بفعل.

دليـــل SUBSCRIPT

وهو رقم صحيح تدل قيمته على موقع عنصر محدد في الجدول.

جـــدول TABLE

وهو مجموعة متتالية من عناصر البيانات تعرف في قسم البيانات بعبارة OCCURS

عنصر الجدول TABLE ELEMENT

وهو عنصر تابع لمجموعة عناصر مكررة تشكل جدول.

قيمة حقيقية TRUTH VALUE

وهي تمثيل لنتيجة تقويم شرط لقيمتين هما :

صـــح TRUE و FALSE خطـــأ

عامل الوحدة UNARY OPERATOR

وهي إشارة الموجب (+) أو إشارة السالب (ــ) التي تسبق متغير أو القوس الأيسر لتعبير رياضي والتي لها تأثير الضرب بـ ١ + أو ١ ــ .

المتغير VARIABLE

وهو حقل يمكن لقيمته أن تتغير أثناء تنفيذ البرنامج ، والمتغير المستخدم في التعبير الرياضي يجب أن يكون حقلاً رقمياً .

فعل VERB

وهي كلمة قد تدل على عمل يجب أن يتخذ بوساطة مترجم برنامج كوبول COBOL COMPILER أو بوساطة برنامج مترجم إلى لغة الآلة .

الكلمة (في لغة كوبول) WORD

سلسلة من الرموز لا تزيد عن ٣٠ رمزاً وتشكل كلمة يستخدمها المبرمج في برنامج كوبول أو اسم نظام أو كلمة محجوزة .

فصل التخزين الداخل للبيانات WORKING-STORAGE SECTION

وهو فصل من قسم البيانات DATA DIVISION والذي يوصف عناصر البيانات المستخدمة كمنطقة عمل في البرنامج ، هذه العناصر يمكن أن تكون عناصر مستقلة أو عناصر بيانات مترابطة ومتفرعة أو يمكن أن تكون الاثنين معاً .

١٥ ــ ٥ حلول مقترحة لبعض التمرينات

```
*          حل تمرين رقم ١    فى الفصل ٣        *
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT16.
 AUTHOR.      KHAYAT-JNAID.

*
*    يقوم هذا البرنامج بقراءة ملف     *
*  بطاقات مثقبة ويكتبه على القرص    *
*                          الممغنط.
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.   IBM-370.
   OBJECT-COMPUTER.   IBM-370.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
       SELECT CARD-F   ASSIGN  TO UT-S-CARDF.
       SELECT CONSMFL ASSIGN  TO UT-S-CONF.
 DATA DIVISION.
 FILE SECTION.
 FD     CARD-F    LABEL RECORD  IS OMITTED.
 01     CARD-REC      PIC     X(80).

*
 FD     CONSMFL   LABEL RECORD  IS STANDARD.
 01     CON-REC       PIC     X(80).
 PROCEDURE DIVISION.
       OPEN  INPUT   CARD-F
             OUTPUT  CONSMFL.

* قراءة سجل من الملف *
 READ-CARD.
       READ CARD-F AT END
            CLOSE CARD-F CONSMFL STOP RUN.

* كتابة سجل على ملف الإخراج *
       MOVE CARD-REC TO CON-REC.
       WRITE CON-REC.
       GO TO F 'AD-CARD.
```

```
*        حل تمرين رقم ٦ في الفصل ٣
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT07.
 AUTHOR.        KHAYAT-JNAID.
*
*   يقوم هذا البرنامج بكتابة اول ٧٠ سجل
*   علي ملف اخراج اول ثم ٣٠ سجل اخر علي
* ملف اخراج ثاني باستخدام PERFORM..TIMES
*
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.    IBM-370.
   OBJECT-COMPUTER.    IBM-370.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
        SELECT INFILE  ASSIGN  TO UT-S-INFILE.
        SELECT DFILE   ASSIGN  TO UT-S-DFILE.
        SELECT TFILE   ASSIGN  TO UT-S-TFILE.
 DATA DIVISION.
 FILE SECTION.
 FD     INFILE     LABEL RECORD  IS OMITTED
                        DATA RECORD IS IN-REC.
 01     IN-REC         PIC      X(80).
*
 FD     DFILE      LABEL RECORD  IS STANDARD
                        DATA RECORD IS  D-REC.
 01     D-REC          PIC      X(80).
*
 FD     TFILE      LABEL RECORD  IS STANDARD
                        DATA RECORD IS  T-REC.
 01     T-REC          PIC      X(80).
 PROCEDURE DIVISION.
        OPEN  INPUT   INFILE
              OUTPUT  DFILE,   TFILE.
*
        PERFORM PAR-1 70 TIMES.
        PERFORM PAR-2 30 TIMES.
        CLOSE INFILE TFILE DFILE STOP RUN.
*
 PAR-1.
        READ INFILE.
```

```
       WRITE D_REC FROM IN_REC.
*
 PAR_2.
       READ INFILE AT END GO TO EXT2.
       WRITE T_REC FROM IN_REC.
 EXT2.
       EXIT.


*
*     ٣ـ٥ في الفصل     ٣ حل تمرين رقم
 IDENTIFICATION DIVISION.
 PROGRAM_ID. TXT05.
 AUTHOR.      KHAYAT_JNAID.
*
* يقوم هذا التمرين بحساب سعر السلعة
*          ويطبع تقريرا بذلك
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE_COMPUTER.   IBM_370.
   OBJECT_COMPUTER.   IBM_370.
  INPUT_OUTPUT SECTION.
  FILE_CONTROL.
       SELECT C_FILE  ASSIGN  TO UT_S_CFILE.
       SELECT P_FILE  ASSIGN  TO UT_S_PFILE.
 DATA DIVISION.
 FILE SECTION.
 FD     C_FILE     LABEL RECORD  IS OMITTED
           BLOCK CONTAINS 0 RECORDS
                       DATA RECORD IS C_REC.
 01     C_REC.
        03     QTY        PIC  9(4)V99.
        03     PRICE      PIC  9(3)V99.
        03     FILLER     PIC  X(69).
*
 FD     P_FILE     LABEL RECORD  IS OMITTED
                       DATA RECORD IS  P_REC.
 01     P_REC          PIC      X(132).
*
 WORKING_STORAGE SECTION.
 01     HDR_1.
```

```
  03    FILLER    PIC X(20)    VALUE SPACES.
  03    FILLER    PIC X(11)    VALUE 'LIST VALUES'.
  03    FILLER    PIC X(101)    VALUE SPACES.
 01     HDR-2.
  03    FILLER    PIC X(03)    VALUE SPACES.
  03    FILLER    PIC X(08)    VALUE 'QUANTITY'.
  03    FILLER    PIC X(06)    VALUE SPACES.
  03    FILLER    PIC X(11)    VALUE '    PRICE'.
  03    FILLER    PIC X(04)    VALUE SPACES.
  03    FILLER    PIC X(28)    VALUE 'AMOUNT'.
 01     DAATA-LINE.
  03    FILLER    PIC X(03)    VALUE SPACES.
  03    P-QNTY    PIC 9(4).99.
  03    FILLER    PIC X(09)    VALUE SPACES.
  03    P-PRICE   PIC 999.99.
  03    FILLER    PIC X(04)    VALUE SPACES.
  03    P-AMONT   PIC Z9(8).99.
  03    FILLER    PIC X(91)    VALUE SPACES.
 01     TOTAL-LINE.
  03    FILLER    PIC X(29)    VALUE '    TOTAL'.
  03    P-TOTAL   PIC ZZZZ9(5).99.
  03    FILLER    PIC X(91)    VALUE SPACES.
*       عدادات تستخدم للتجميع
 01     TOTAL-AREAS.
  03    AMONT-A     PIC 9(9)V99 VALUE ZERO.
  03    TOTAL-A     PIC 9(9)V99 VALUE ZEROS.
  03    LINE-CNTR PIC 99        VALUE 0.
 PROCEDURE DIVISION.
        OPEN    INPUT    C-FILE
                OUTPUT   P-FILE.
*                 طباعة العناوين
   PRINT-HEADERS.
        MOVE    SPACES TO     P-REC.
        WRITE   P-REC  AFTER ADVANCING  PAGE.
        MOVE    HDR-1  TO     P-REC.
        WRITE   P-REC  AFTER  ADVANCING 3 LINES.
        MOVE    HDR-2  TO     P-REC.
        WRITE   P-REC  AFTER  ADVANCING 2 LINES.
        MOVE    SPACES TO     P-REC.
        WRITE   P-REC  AFTER 1 LINE.
        MOVE    ZERO   TO     LINE-CNTR.
```

```
MAIN-LOOP.
      READ    C-FILE AT END
                          GO TO TOTAL-P.
      MULTIPLY QTY BY PRICE GIVING AMONT-A.
      ADD     AMONT-A     TO    TOTAL-A.
      MOVE    AMONT-A     TO    P-AMONT.
      MOVE    QTY         TO    P-QNTY.
      MOVE    PRICE       TO    P-PRICE.
      MOVE    DAATA-LINE TO     P-REC.
      WRITE   P-REC      AFTER ADVANCING 1 LINE.
      ADD     1          TO     LINE-CNTR.
      IF      LINE-CNTR =       24
              GO TO      PRINT-HEADERS
      ELSE
              GO TO  MAIN-LOOP.
TOTAL-P.
      MOVE    TOTAL-A     TO    P-TOTAL.
      MOVE    TOTAL-LINE TO     P-REC.
      WRITE   P-REC       AFTER 3.
*
      CLOSE   C-FILE,     P-FILE.
      STOP    RUN.
```

```
                          LIST VALUES
     QUANTITY             PRICE            AMOUNT

     0020.00              020.00           00000400.00
     0034.00              012.00           00000408.00
     0067.00              023.00           00001541.00
     0050.00              034.00           00001700.00
     0035.50              023.30           00000827.15
     0025.50              045.50           00001160.25

     TOTAL                                 06036.40
```

شكل المخرجات

```
*     حل تمرين رقم ١  في الفصل ١١-٤     *
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT11.
 AUTHOR.      KHAYAT-JNAID.
*
* يقوم هذا البرنامج بانشاء ملف مسلسل *
* 'DISPLAY' &'ACCEPT'باستخدام تعليمتي
*
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.   IBM-370.
   OBJECT-COMPUTER.   IBM-370.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
       SELECT PAYFILE ASSIGN  TO UT-S-PFILE.
 DATA DIVISION.
 FILE SECTION.
 FD     PAYFILE   LABEL RECORD  IS STANDARD
                     DATA RECORD IS PAY-REC.
 01     PAY-REC.
        03     EM-NO     PIC  9(4).
        03     EM-NAME   PIC  A(20).
        03     EM-SAL    PIC  9(5)V99.
        03     EM-TRANS  PIC  9(3)V99.
        03     BL-NK     PIC  X(4).
*
 PROCEDURE DIVISION.
        OPEN   OUTPUT  PAYFILE.
*
  AC-NUMB.
        DISPLAY '     أدخل رقم الموظف:'.
        ACCEPT  EM-NO.
        IF  EM-NO  =   9999
                   CLOSE PAYFILE STOP RUN.
        IF  EM-NO  NOT NUMERIC OR
            EM-NO  LESS THAN 1 GO TO AC-NUMB.
  AC-NAME.
        DISPLAY '     أدخل اسم الموظف:'.
        ACCEPT  EM-NAME.
        IF EM-NAME  =   SPACE OR
            EM-NAME  NOT ALPHABETIC
                            GO TO AC-NAME.
```

```
 AC-SAL.
      DISPLAY '     ادخل راتب الموظف:'.
      ACCEPT  EM-SAL.
      IF EM-SAL NOT NUMERIC GO TO AC-SAL.
 AC-TRANSP.
      DISPLAY '     ادخل بدل النقل:'.
      ACCEPT  EM-TRANS.
      IF EM-TRANS NOT NUMERIC
                        GO TO AC-TRANSP.
*
      WRITE   PAY-REC.
      GO TO   AC-NUMB.



*  حل تمرين رقم ٣   في الفصل ٤-١١  *
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT10.
 AUTHOR.      KHAYAT-JNAID.
*
*  يقوم هذا البرنامج بحساب صافي الراتب  *
*                              وطباعته.
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.    IBM-370.
   OBJECT-COMPUTER.    IBM-370.
   SPECIAL-NAMES.      DECIMAL-POINT IS COMMA.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
      SELECT PAYFILE ASSIGN  TO UT-S-PFILE.
      SELECT P-FILE  ASSIGN  TO UT-S-PRFL.
 DATA DIVISION.
 FILE SECTION.
 FD    PAYFILE   LABEL RECORD  IS STANDARD
         BLOCK CONTAINS 0 RECORDS
                    DATA RECORD IS PAY-REC.
 01    PAY-REC.
       03     EM-NO     PIC  9(4).
       03     EM-NAME   PIC  X(20).
       03     EM-SAL    PIC  9(5)V99.
       03     EM-TRANS  PIC  9(3)V99.
       03     FILLER    PIC  X(44).
```

```
*
 FD     P-FILE     LABEL RECORD  IS OMITTED
                           DATA RECORD IS  P-REC.
 01     P-REC          PIC      X(132).
*
 WORKING-STORAGE SECTION.
 01     HDR-1.
  03    FILLER    PIC     X(56)    VALUE SPACES.
  03    FILLER    PIC     X(19)    VALUE
                         'BRIGHT STAR COMPANY'.
  03    FILLER    PIC     X(57)    VALUE SPACES.
 01     HDR-2.
  03    FILLER    PIC     X(10)    VALUE
                                    ' PAGE NO :'.
  03    P-CTR     PIC     ZZ9.
  03    FILLER    PIC     X(44)    VALUE SPACES.
  03    FILLER    PIC     X(15)    VALUE
                          ' PAYROLL REPORT'.
  03    FILLER    PIC     X(44)    VALUE SPACES.
  03    FILLER    PIC     X(06)    VALUE 'DATE:'.
  03    DAY-P     PIC     99.
  03    FILLER    PIC     X        VALUE '/'.
  03    MONTH-P   PIC     99.
  03    FILLER    PIC     XXX      VALUE '/19'.
  03    YEAR-P    PIC     99.
  03    FILLER    PIC     X(3)     VALUE SPACES.
 01     HDR-3.
  03    FILLER    PIC     X(20)
                        VALUE '  EMPLOYEE NO.'.
  03    FILLER    PIC     X(20)
                        VALUE 'EMPLOYEE  NAME'.
  03    FILLER    PIC     X(14)
                        VALUE 'BASE SALARY'.
  03    FILLER    PIC     X(16)
                        VALUE 'TRANSPORTATION'.
  03    FILLER    PIC     X(15)
                        VALUE 'SOCIAL SECURITY'.
  03    FILLER    PIC     X(13)
                         VALUE '  SOCIAL-FUND'.
  03    FILLER    PIC     X(13)
                         VALUE '      NET PAY'.
```

```
 03    FILLER    PIC      X(22)
                       VALUE '      SIGNATURE'.
01     DATA-LINE.
 03    FILLER    PIC      X(06)    VALUE SPACES.
 03    P-NO      PIC      ZZZ9.
 03    FILLER    PIC      X(06)    VALUE SPACES.
 03    P-NAME    PIC      X(20).
 03    FILLER    PIC      X(04)    VALUE SPACES.
 03    P-SAL     PIC      Z(4)9,99.
 03    FILLER    PIC      X(08)    VALUE SPACES.
 03    P-TRANS   PIC      ZZ9,99.
 03    FILLER    PIC      X(09)    VALUE SPACES.
 03    P-SOCIAL  PIC      Z(3)9,99.
 03    FILLER    PIC      X(10)    VALUE SPACES.
 03    P-FUNDS   PIC      Z(3)9,99.
 03    FILLER    PIC      X(07)    VALUE SPACES.
 03    P-NET     PIC      Z(5)9,99.
 03    FILLER    PIC      X(22)    VALUE SPACES.
01     TOTAL-LINE.
 03    FILLER    PIC      X(29) VALUE '    TOTAL'.
 03    P-TOTAL   PIC      9(4).99.
 03    FILLER    PIC      X(91)    VALUE SPACES.
*  لقبول التاريخ من الحاسب
01     CURENT-DATE.
 03    C-YEAR    PIC      99.
 03    C-MONTH   PIC      99.
 03    C-DAY     PIC      99.
*      عدادات تستخدم للتجميع
77     SOCIAL    PIC      9(4)V99.
77     FUNDS     PIC      9(4)V99.
77     NETPAY    PIC      9(6)V99.
77     L-CON     PIC      99        VALUE 0.
77     P-CNTR    PIC      999       VALUE 0.
PROCEDURE DIVISION.
      OPEN      INPUT    PAYFILE
                 OUTPUT   P-FILE.
*  ACCEPT DATE & PRINT HEADERS.
      ACCEPT    CURENT-DATE FROM DATE.
      MOVE      C-YEAR         TO    YEAR-P.
      MOVE      C-MONTH        TO    MONTH-P.
      MOVE      C-DAY          TO    DAY-P.
      PERFORM   NEW-PAGE.
```

```
*
 MAIN-LOOP.
       READ      PAYFILE      AT END
                 CLOSE PAYFILE P-FILE STOP RUN.

*
*   حساب التامينات الاجتماعية :
      MULTIPLY EM-SAL BY 0,05 GIVING SOCIAL.

*
*
*   معالجة حالة الراتب اقل من ١٠٠٠,٠١ ريال :
      IF          EM-SAL      <      1000,01
       MULTIPLY EM-SAL BY 0,01 GIVING FUNDS
       ELSE

*
*
*   معالجة حالة الراتب بين ١٠٠٠ و ٢٠٠٠ ريالا :
       IF EM-SAL > 1000,01 AND EM-SAL < 2000,01
          COMPUTE  FUNDS   =
             ( EM-SAL - 1000 ) * 0,03 + 10
       ELSE

*
*   معالجة حالة الراتب اكبر من ٢٠٠٠,٠٠ ريال :
*   ملاحظة : تعتبر ١٠ و ٣٠ هم حاصل النسبة عن
*   الالفين الاولين .
        COMPUTE  FUNDS    =
           ( EM-SAL - 2000 ) * 0,05 + 10 + 30.

*
*
*   حساب صافي الراتب :
       COMPUTE NETPAY        =
        EM-SAL + EM-TRANS - ( FUNDS + SOCIAL ).

*
*
*   تحريك الحقول والنتائج الى سطر الطباعة :
       MOVE     EM-NO       TO      P-NO.
       MOVE     EM-NAME     TO      P-NAME.
```

```
       MOVE      EM-SAL       TO       P-SAL.
       MOVE      EM-TRANS     TO       P-TRANS.
       MOVE      SOCIAL       TO       P-SOCIAL.
       MOVE      FUNDS        TO       P-FUNDS.
       MOVE      NETPAY       TO       P-NET.


*
*
*  :  كتابة السجل على ملف الطباعة
       WRITE     P-REC        FROM     DATA-LINE
                 AFTER        ADVANCING 2 LINES.
       ADD       1            TO       L-CON.
       IF        L-CON        >        25
                              PERFORM NEW-PAGE.
       GO TO     MAIN-LOOP.


*
*
*  : فقرة قلب الصفحة وكتابة العناوين
 NEW-PAGE.
       MOVE      ZERO         TO       L-CON.
       ADD       1            TO       P-CNTR.
       MOVE      P-CNTR       TO       P-CTR.
       MOVE      SPACES       TO       P-REC.
       WRITE P-REC  AFTER  PAGE.
       WRITE P-REC  FROM HDR-1 AFTER 1 LINE.
       WRITE P-REC  FROM HDR-2 AFTER 1 LINE.
       WRITE P-REC  FROM HDR-3 AFTER 2 LINE.
  EXTP.
       EXIT.
```

أنظر شكل المخرجات صفحة ٥٩٩

```
*  ١١-٤ حل تمرين رقم ٣ في الفصل  *
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT08.
 AUTHOR.        KHAYAT-JNAID.
*
*    يقوم هذا البرنامج بطباعة درجة وتقدير
*    الطالب مستخدما CONDITION NAME
*
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.    IBM-370.
   OBJECT-COMPUTER.    IBM-370.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
       SELECT STDFILE ASSIGN  TO UT-S-STDF.
       SELECT PRINTF  ASSIGN  TO UT-S-PRFL.
 DATA DIVISION.
 FILE SECTION.
 FD      STDFILE  LABEL RECORD  IS OMITTED
  BLOCK CONTAINS 0 RECORDS.
 01      STD-REC.
  03     STD-NAME       PIC       X(25).
  03     STD-MARK       PIC       999.
    88 FAIL        VALUES  ARE    00    THRU 59.
    88 PASS        VALUES  ARE    60    THRU 69.
    88 GOOD        VALUES  ARE    70    THRU 79.
    88 VRYGOOD     VALUES  ARE    80    THRU 89.
    88 EXCELNT     VALUES  ARE    90    THRU 100.
  03     FILLER         PIC       X(52).
*
 FD      PRINTF     LABEL   RECORD  IS OMITTED.
 01      P-REC.
  03     FILLER         PIC       X(3).
  03     S-NAME         PIC       X(25).
  03     FILLER         PIC       X(6).
  03     S-MARK         PIC       9(3).
  03     FILLER         PIC       X(4).
  03     S-GRAD         PIC       X(7).
  03     FILLER         PIC       X(84).
 WORKING-STORAGE     SECTION.
 77      END-OF-JOB     PIC       XXX   VALUE 'NO'.
```

```
PROCEDURE DIVISION.
       OPEN     INPUT   STDFILE
                OUTPUT  PRINTF.
* قراءة سجل من الملف
       READ     STDFILE     AT     END
                  MOVE 'YES' TO END-OF-JOB.
       PERFORM WRITE-READ THRU  EXT
                  UNTIL END-OF-JOB = 'YES'.
       CLOSE STDFILE, PRINTF.
       STOP  RUN.
* كتابة سجل على ملف الإخراج و قراءة سجل جديد
 WRITE-READ.
     IF  FAIL     MOVE    'FAIL'     TO  S-GRAD.
     IF  PASS     MOVE    'PASS'     TO  S-GRAD.
     IF  GOOD     MOVE    'GOOD'     TO  S-GRAD.
     IF  VRYGOOD MOVE     'V.GOOD'   TO  S-GRAD.
     IF  EXCELNT MOVE     'EXCELL.'  TO  S-GRAD.
       MOVE  STD-NAME  TO      S-NAME.
       MOVE  STD-MARK  TO      S-MARK.
       WRITE P-REC     AFTER  2.
       READ  STDFILE   AT      END
                  MOVE   'YES' TO END-OF-JOB.
EXT.
      EXIT.
```

شكل المخرجات

```
ZUHAIR KHAYAT                         099    EXCELL.

SAMI ALHALABI                         090    EXCELL.

MOJAHED ALHARBI                       050    FAIL

BADR ALJUNDI                          055    FAIL

SUBHI RAJAB                           073    GOOD

AHMAD HAMEED GHAZI                    078    GOOD

FERAS AHMED JNAID                     084    V.GOOD
```

KHALDOON AHMED	085	V.GOOD
SAMEER HUSNI	070	GOOD
MANSUR AL GHADAN	060	PASS
ABDULAZIZ QUAIZ	063	PASS
SALEM KHANAWATI	065	PASS
BASHIR ZUNDAH	069	PASS
ALI AL YUSEF	057	FAIL
MOHAMED ZAKI	053	FAIL
MOLHAM ZAINI	089	V.GOOD
AHMED JAMALI	099	EXCELL.
ABDULRAHMAN HARBI	098	EXCELL.
ABDUL AZIZ MANSOOR	064	PASS
GHIAS ALCHATTA	065	PASS
HUSNI AL SHMEMRI	087	V.GOOD
MONEER QUATARI	085	V.GOOD
NASER AL OHIBI	084	V.GOOD
OTHMAN AL BASHEER	080	V.GOOD
ALMAJED YAHYAWI	083	V.GOOD
MAAMAR MAKID	080	V.GOOD
LINAH KHAYAT	095	EXCELL.
YOUSUF QABOLI	014	FAIL

```
*     حل تمرين رقم ١ في الفصل ٥-١٠
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT09.
 AUTHOR.      KHAYAT-JNAID.
*
*يقوم هذا البرنامج بعمل تحقق على
*     ملف الإدخال ويطبع قائمه بالاخطاء
*
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.   IBM-370.
   OBJECT-COMPUTER.   IBM-370.
  INPUT-OUTPUT SECTION.
   FILE-CONTROL.
       SELECT SUBFILE ASSIGN  TO UT-S-SFIL.
       SELECT P-FILE  ASSIGN  TO UT-S-PFIL.
       SELECT INVFILE ASSIGN  TO UT-S-IFIL.
       SELECT VALFILE ASSIGN  TO UT-S-VFIL.
 DATA DIVISION.
 FILE SECTION.
*
* : توصيف ملف الإدخال
 FD     SUBFILE   LABEL RECORD  IS STANDARD
            BLOCK CONTAINS 0 RECORDS
                       DATA RECORD IS SUB-REC.
 01     SUB-REC.
        03  D-NO        PIC  9(5).
        03  D-NAME      PIC  X(20).
        03  D-ADRESS    PIC  X(24).
        03  D-DATE.
         05 D-DAY   PIC  9(2).
         88 V-DAY   VALUES ARE 01   THRU   30.
         05 D-MNTH  PIC   9(2).
         88 V-MNTH  VALUES ARE  01  THRU   12.
         05 D-YEAR  PIC  9(4).
         88 V-YEAR  VALUES ARE 1402 THRU 1410.
        03  D-CODE     PIC  99.
        88  V-CODE  VALUES ARE  01  THRU   10.
        03  D-AREA     PIC  99.
        88  V-AREA  VALUES ARE  01  THRU   15.
        03  D-CNTR     PIC  9(7).
```

```
        03  D-LREAD      PIC  9(7).
        03  FILLER       PIC  X(5).
*
* : توصيف ملف الطباعة
 FD     P-FILE     LABEL RECORD  IS OMITTED
                            DATA RECORD IS  P-REC.
 01     P-REC           PIC      X(132).
*
*
* :  توصيف ملف السجلات الخطا
 FD     INVFILE    LABEL RECORD  IS STANDARD
                        DATA RECORD IS  INV-REC.
 01     INV-REC         PIC      X(080).
*
*
* : توصيف ملف السجلات الصحيحة
 FD     VALFILE    LABEL RECORD  STANDARD
                    DATA RECORD IS  VAL-REC.
 01     VAL-REC            PIC      X(080).
*
 WORKING-STORAGE SECTION.
*
 01     HDR-1.
  03    FILLER   PIC    X(58)   VALUE SPACES.
  03    FILLER   PIC    X(16)   VALUE
        'ELECTRIC COMPANY'.
  03    FILLER   PIC    X(58)   VALUE SPACES.
*
 01     HDR-2.
  03    FILLER   PIC    X(12)   VALUE
        '        DATE:'.
  03    DAY-P    PIC    99.
  03    FILLER   PIC    X       VALUE '/'.
  03    MNTH-P   PIC    99.
  03    FILLER   PIC    XXX     VALUE '/19'.
  03    YEAR-P   PIC    99.
  03    FILLER   PIC    X(29)   VALUE SPACES.
  03    FILLER   PIC    X(34)   VALUE
       'LIST OF INVALID SUBSCRIBER RECORDS'.
  03    FILLER   PIC    X(34)   VALUE SPACES.
  03    FILLER   PIC    X(09)   VALUE
       'PAGE NO :'.
```

```
  03    P-CTR     PIC     ZZ9.
*
 01     HDR-3.
  03    FILLER    PIC     X(20)    VALUE
        ' SUBSCRIBER NO.'.
  03    FILLER    PIC     X(20)    VALUE
        'SUBS. N A M E '.
  03    FILLER    PIC     X(27)    VALUE
        'SUBS. A D D R E S S'.
  03    FILLER    PIC     X(20)    VALUE
        'SUBSCRIPTION DATE'.
  03    FILLER    PIC     X(13)    VALUE
        ' SUB. C O D E'.
  03    FILLER    PIC     X(12)    VALUE
        '    SUB. AREA'.
  03    FILLER    PIC     X(20)    VALUE
        '  LAST READING'.
*
 01     DETAL-L.
  03    FILLER    PIC     X(04)    VALUE SPACES.
  03    D-NO      PIC     XXXXX.
  03    FILLER    PIC     X(07)    VALUE SPACES.
  03    D-NAME    PIC     X(20).
  03    FILLER    PIC     X(03)    VALUE SPACES.
  03    D-ADRESS  PIC     X(24).
  03    FILLER    PIC     X(10)    VALUE SPACES.
  03    D-DATE.
      05 D-YEAR   PIC     9(4).
      05 FILLER   PIC     X        VALUE '/'.
      05 D-MNTH   PIC     9(2).
      05 FILLER   PIC     X        VALUE '/'.
      05 D-DAY    PIC     9(2).
  03    FILLER    PIC     X(05)    VALUE SPACES.
  03    D-CODE    PIC     99.
  03    FILLER    PIC     X(13)    VALUE SPACES.
  03    D-AREA    PIC     99.
  03    FILLER    PIC     X(09)    VALUE SPACES.
  03    D-CNTR    PIC     Z(5)99.
  03    FILLER    PIC     X        VALUE SPACES.
  03    D-LREAD   PIC     Z(5)99.
  03    FILLER    PIC     X        VALUE SPACES.
```

```
*
 01      AST-L.
   03    FILLER    PIC    X(06)    VALUE SPACES.
   03    AST-1     PIC    X.
   03    FILLER    PIC    X(18)    VALUE SPACES.
   03    AST-2     PIC    X.
   03    FILLER    PIC    X(24)    VALUE SPACES.
   03    AST-3     PIC    X.
   03    FILLER    PIC    X(22)    VALUE SPACES.
   03    AST-4     PIC    X.
   03    FILLER    PIC    X(04)    VALUE SPACES.
   03    AST-5     PIC    X.
   03    FILLER    PIC    X(03)    VALUE SPACES.
   03    AST-6     PIC    X.
   03    FILLER    PIC    X(07)    VALUE SPACES.
   03    AST-7     PIC    X.
   03    FILLER    PIC    X(12)    VALUE SPACES.
   03    AST-8     PIC    X.
   03    FILLER    PIC    X(09)    VALUE SPACES.
   03    AST-9     PIC    X.
   03    FILLER    PIC    X(08)    VALUE SPACES.
   03    AST-10    PIC    X.
   03    FILLER    PIC    X(03)    VALUE SPACES.
*
* لقبول التاريخ من الحاسب :
 01      CURENT-DATE.
   03    C-YEAR    PIC    99.
   03    C-MNTH    PIC    99.
   03    C-DAY     PIC    99.
*
* الحقل التالي يستخدم لاختبار نهاية الملف :
 77      EOFL      PIC    XXX      VALUE SPACES.
*
* عدادات تستخدم للتجميع :
 77      LINES-CONT PIC   99       VALUE 25.
 77      FLAG       PIC   9        VALUE ZERO.
 77      PAGE-CNTR  PIC   99       VALUE ZERO.
 77      LINE-NO    PIC   9999     VALUE 0.
*
 PROCEDURE DIVISION.
      OPEN    INPUT    SUBFILE
              OUTPUT   P-FILE, INVFILE VALFILE.
```

```
*
*       قبول التاريخ وطباعة العناوين
*        وقراءة السجل الاول من الملف
*
       ACCEPT   CURENT-DATE FROM DATE.
       MOVE     C-YEAR        TO   YEAR-P.
       MOVE     C-MNTH        TO   MNTH-P.
       MOVE     C-DAY         TO   DAY-P.
       PERFORM  NEW-PAGE      THRU EXTP.
       READ     SUBFILE       AT END
                MOVE         'END' TO EOFL.
*
*  المعالجة الرئيسية للبرنامج
       PERFORM  MAIN-LOOP  THRU   END-LOOP
                    UNTIL       EOFL = 'END'.
       PERFORM  END-RTN       THRU   EXT-END.
       STOP     RUN.
*
*
*      عمل تحقق علي كل حقل وتحريك نجمة
*    تحت الحقل غير الصحيح :
*
 MAIN-LOOP.
        MOVE      ZERO            TO FLAG.
        MOVE      SPACES          TO AST-L.
        IF D-NO      OF SUB-REC NOT NUMERIC OR
           D-NO      OF SUB-REC <    1
                                MOVE '*' TO AST-1.
*
        IF D-NAME   OF SUB-REC = SPACE OR
           D-NAME   OF SUB-REC NUMERIC
                                MOVE '*' TO AST-2.
*
        IF D-ADRESS    OF SUB-REC = SPACE
                                MOVE '*' TO AST-3.
*
        IF      NOT V-YEAR OF SUB-REC
                                MOVE '*' TO AST-4.
*
        IF      NOT V-MNTH OF SUB-REC
                                MOVE '*' TO AST-5.
```

```
*
       IF     NOT V-DAY  OF SUB-REC
                              MOVE '*' TO AST-6.
*
       IF     NOT V-CODE OF SUB-REC
                              MOVE '*' TO AST-7.
*
       IF NOT V-AREA OF SUB-REC
                              MOVE '*' TO AST-8.
*
       IF D-CNTR      OF SUB-REC NOT NUMERIC
       OR D-CNTR   OF SUB-REC LESS    THAN 1
                              MOVE '*' TO AST-9.
*
       IF D-LREAD     OF SUB-REC NOT NUMERIC
                              MOVE '*' TO AST-10.
*
        IF AST-L = SPACES
           WRITE  VAL-REC    FROM SUB-REC
                              MOVE  1  TO FLAG.
       IF FLAG  = 1           NEXT SENTENCE
       ELSE
       IF LINES-CONT > 24
                 PERFORM NEW-PAGE THRU EXTP
       ELSE
         ADD   1          TO LINES-CONT
         MOVE CORRESPONDING SUB-REC
                                 TO DETAL-L
         WRITE P-REC      FROM DETAL-L AFTER 1
         WRITE P-REC      FROM AST-L   AFTER 1
         WRITE INV-REC FROM  SUB-REC.
       READ  SUBFILE      AT END
                          MOVE 'END ' TO EOFL.
 END-LOOP.
       EXIT.
*
*   فقرة قلب الصفحة وطبع العناوين :
 NEW-PAGE.
       MOVE  SPACES       TO      P-REC.
       WRITE P-REC        AFTER   PAGE.
       MOVE  ZERO         TO      LINES-CONT.
       ADD   1            TO      PAGE-CNTR.
```

```
       MOVE  PAGE-CNTR  TO     P-CTR.
       WRITE P-REC FROM HDR-1 AFTER 1 LINES.
       WRITE P-REC FROM HDR-2 AFTER 1 LINES.
       WRITE P-REC FROM HDR-3 AFTER 1 LINES.
 EXTP.
       EXIT.
*
*
*    : فقرة نهاية المعالجة
 END-RTN.
     CLOSE SUBFILE VALFILE, INVFILE, P-FILE.
 EXT-END.
       EXIT.
```

أنظر شكل المخرجات صفحة ٦٠٠

●

```
*    حل تمرين رقم ٦   في الفصل ٥-١٠
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT12.
 AUTHOR.       KHAYAT-JNAID.
*
*      يقوم هذا البرنامج باستدعاء برنامج
* باستدعاء برنامج فرعي اخر وذلك لاحتساب
*                              قسط الراتب
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.    IBM-370.
   OBJECT-COMPUTER.    IBM-370.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
       SELECT CARD-F   ASSIGN   TO UT-S-CARD.
       SELECT PRINTF   ASSIGN  TO UT-S-PRFL.
 DATA DIVISION.
 FILE SECTION.
 FD     CARD-F       LABEL RECORD  IS OMITTED
                      DATA RECORD IS CARD-REC.
 01     CARD-REC.
        03     EM-NO     PIC  9(5).
        03     EM-NAME   PIC  X(30).
```

```
          03     EM-SAL    PIC  9(5)V99.
          03     EM-TRANS  PIC  9(3)V99.
          03     EM-TRAIN  PIC  9(5)V99.
          03     EM-DATE.
           05    EM-DAY    PIC  99.
           05    EM-MNTH   PIC  99.
           05    EM-YEAR   PIC  9999.
          03     FILLER    PIC  X(18).
*
 FD       PRINTF    LABEL RECORD  IS OMITTED
                         DATA RECORD IS  P-REC.
 01       P-REC          PIC     X(132).
*
 WORKING-STORAGE SECTION.
*   الحقول التاليه ستذهب الى البرنامج
*   الفرعي بتعليمه CALL ليتم الحساب بها
 01       CALC.
          03    W-SAL       PIC  9(5)V99.
          03    W-TRANS     PIC  9(3)V99.
          03    W-TRAIN     PIC  9(5)V99.
          03    W-NET       PIC  9(5)V99.
*
 01       D-L.
          03    FILLER PIC X(05)    VALUE SPACES.
          03    P-NO   PIC 9(5).
          03    FILLER PIC X(05)    VALUE SPACES.
          03    P-NAME PIC X(30).
          03    FILLER PIC X(05)    VALUE SPACES.
          03    P-SAL  PIC Z(4)9.99.
          03    FILLER PIC X(05)    VALUE SPACES.
          03    P-TRAN PIC ZZ9.99.
          03    FILLER PIC X(09)    VALUE SPACES.
          03    P-TRIN PIC Z(4)9.99.
          03    FILLER PIC X(05)    VALUE SPACES.
          03    P-NET  PIC Z(4)9.99.
          03    FILLER PIC X(05)    VALUE SPACES.
 PROCEDURE DIVISION.
        OPEN      INPUT   CARD-F
                   OUTPUT  PRINTF.
*
 MAIN-PROG.
        READ      CARD-F        AT END
                  CLOSE CARD-F PRINTF STOP RUN.
```

```
*
*  CALC ارسال السجل المقروء مع مساحات العمل
*       : الى البرنامج الفرعي SUBPGM للحساب
*
     CALL 'TXTSUB'  USING CARD-REC CALC.
*
*      بعد تنفيذ التعليمة اعلاه تاتي الحقول
*      لتحرك الى سطر الطباعة
      MOVE      EM-NO       TO        P-NO.
      MOVE      EM-NAME     TO        P-NAME
      MOVE      W-SAL       TO        P-SAL.
      MOVE      W-TRANS     TO        P-TRAN.
      MOVE      W-TRAIN     TO        P-TRIN.
      MOVE      W-NET       TO        P-NET.
*
      WRITE     P-REC       FROM      D-L   AFTER 2.
      GO TO     MAIN-PROG.
```

- ```
*   حل تمرين رقم ٣   في الفصل ٥-١٠
*   تابع لحل البرنامج TXT12
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXTSUB.
 AUTHOR.   KHAYAT-JNAID.
*
* هذا البرنامج المستدعى من قبل البرنامج *
*   TXT12 ويقوم باحتساب قسط الراتب
*
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.   IBM-370.
   OBJECT-COMPUTER.   IBM-370.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 77   W-DAYS   PIC 99.
 LINKAGE   SECTION.
*   الحقول التاليه تستقبل مايقابلها من
* CALL حقول مررت لهذا البرنامج بتعليمه
*   LINKAGE SECTION   عبر قسم
 01   L-REC.
       03   L-NO   PIC 9(5).
       03   L-NAME   PIC X(30).
```
```

```
           03        L-SAL    PIC   9(5)V99.
           03        L-TRANS  PIC   9(3)V99.
           03        L-TRAIN  PIC   9(5)V99.
           03        L-DATE.
            05       L-DAY    PIC   99.
            05       L-MNTH   PIC   99.
            05       L-YEAR   PIC   9999.
           03       FILLER    PIC   X(18).
 01        WS-CALC.
           03     WS-SAL        PIC  9(5)V99.
           03     WS-TRANS      PIC  9(3)V99.
           03     WS-TRAIN      PIC  9(5)V99.
           03     WS-NET        PIC  9(5)V99.
*
* : USING لاحظ تغير عنوان القسم التالي واستخدام
 PROCEDURE DIVISION USING L-REC WS-CALC.
*
          SUBTRACT L-DAY FROM 30 GIVING  W-DAYS.
          COMPUTE  WS-SAL      ROUNDED =
                         L-SAL       / 30 * W-DAYS.
          COMPUTE  WS-TRANS  ROUNDED =
                         L-TRANS    / 30 * W-DAYS.
          COMPUTE  WS-TRAIN  ROUNDED =
                         L-TRAIN    / 30 * W-DAYS.
          ADD          WS-SAL
                       WS-TRANS
                       WS-TRAIN
          GIVING       WS-NET.
 RETURN-TO-MAIN.
          EXIT PROGRAM.
*         العودة للبرنامج الرئيسي TXT12
*
```

```
*         حل تمرين رقم ١ في الفصل ٧-٤
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT14.
 AUTHOR.      KHAYAT-JNAID.
*
*    يقوم هذا البرنامج بدمج ثلاث ملفات
*    ادخال بملف اخراج واحد    ثم تتم
*  معالجة الملف الناتج وطباعة محتواه
```

```
*
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.   IBM-370.
   OBJECT-COMPUTER.   IBM-370.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
       SELECT INF1    ASSIGN  TO UT-S-INF1.
       SELECT INF2    ASSIGN  TO UT-S-INF2.
       SELECT INF3    ASSIGN  TO UT-S-INF3.
       SELECT MRGF    ASSIGN  TO UT-S-MRGF.
       SELECT OTMRG   ASSIGN  TO UT-S-OTMRG.
       SELECT PRINTF  ASSIGN  TO UT-S-PRFL.
 DATA DIVISION.
 FILE  SECTION.
*
 FD     INF1  LABEL  RECORD  IS STANDARD
            RECORD  CONTAINS 80 CHARACTERS
                       DATA RECORD IS IN-REC1.
 01     IN-REC1          PIC  X(80).
*
 FD     INF2  LABEL  RECORD  IS STANDARD
             RECORD  CONTAINS 80 CHARACTERS
                       DATA RECORD IS IN-REC2.
 01     IN-REC2          PIC  X(80).
*
 FD     INF3  LABEL  RECORD  IS STANDARD
            RECORD  CONTAINS 80 CHARACTERS
                      DATA RECORD IS IN-REC3.
 01     IN-REC3          PIC  X(80).
*
 SD     MRGF
            RECORD  CONTAINS 80 CHARACTERS
                     DATA RECORD IS MRG-REC.
 01     MRG-REC.
  03 MRG-NO         PIC    9(6).
  03 MRG-DATA       PIC    X(74).
*
 FD     OTMRG LABEL RECORD  STANDARD
             RECORD  CONTAINS 80 CHARACTERS
                      DATA RECORD IS OUT-REC.
```

```
 01     OUT_REC.
  03 CST_NO          PIC     9(6).
  03 CST_NAME        PIC     X(24).
  03 CST_BAL         PIC     S9(8)V99.
  03 CST_FLRE        PIC     X(40).
*
 FD     PRINTF     LABEL RECORD  IS OMITTED
                         DATA RECORD IS  P_REC.
 01     P_REC           PIC      X(132).
*
 WORKING_STORAGE SECTION.
 01     HDR1.
  03    FILLER  PIC     X(57)   VALUE SPACES.
  03    FILLER  PIC     X(16)   VALUE
                         'CUSTOMMER REPORT'.
  03    FILLER  PIC     X(57)   VALUE SPACES.
 01     HDR2.
  03    FILLER  PIC     X(08)   VALUE
                                 ' DATE :'.
  03    P_DATE  PIC     99/99/99.
  03    FILLER  PIC     X(100)  VALUE SPACES.
  03    FILLER  PIC     X(07)   VALUE
                                 'PAGE :'.
  03    P_PAGE  PIC     ZZ9.
  03    FILLER  PIC     X(5)    VALUE SPACES.
 01     HDR3.
  03    FILLER PIC      X(40)   VALUE
                        '      CUSTOMMER NO.'.
  03    FILLER PIC      X(21)   VALUE
                       '      CUSTOMMER NAME'.
  03    FILLER PIC      X(21)   VALUE
                       '              BALANCE'.
  03    FILLER PIC      X(50)   VALUE
                      '              COMMENTS'.
*
 01     DATA_LINE.
  03    FILLER  PIC     X(06)   VALUE SPACES.
  03    P_NO    PIC     ZZZZ99.
  03    FILLER  PIC     X(26)   VALUE SPACES.
  03    P_NAME  PIC     X(24).
  03    FILLER  PIC     X(15)   VALUE SPACES.
```

```
   03    P_BAL    PIC      ++++++999.99.
   03    FILLER   PIC      X(27)    VALUE SPACES.
   03    P_COMNT  PIC      X(7).
   03    FILLER   PIC      X(10)    VALUE SPACES.

*
*          حقل عمل لقبول التاريخ
*
 01     C_DATE   PIC      9(6).
 01     PAG_CTR  PIC      999      VALUE ZERO.
 01     LIN_CTR  PIC      99       VALUE 25.
 77     FLAG     PIC      9        VALUE ZERO.

*
 PROCEDURE DIVISION.
*  تعليمة دمج الملفات الثلاث في ملف
*                         اخراج واحد.
*.OPEN  لاحظ ان الملفات لم تفتح بتعليمة
*
       MERGE  MRGF
              ASCENDING KEY MRG_NO
       USING  INF1 INF2 INF3
              GIVING    OTMRG.
*  بعد تنفيذ تعليمة الدمج السابقة
*  تبدأ معالجة الملف OTMRG معالجة
*                         عادية.
*
 START_PROCESSING.
      OPEN INPUT OTMRG
           OUTPUT   PRINTF.
      ACCEPT C_DATE FROM DATE.
      MOVE C_DATE TO P_DATE.
        READ OTMRG AT END
              MOVE  1   TO   FLAG.

*
 MAIN_LOOP.
       PERFORM MOVE_WRITE
                  THRU EXT_MOVE_WRITE
        UNTIL FLAG EQUAL 1.
       CLOSE OTMRG PRINTF.
       STOP RUN.
```

```
*
 MOVE-WRITE.
        IF       CST-BAL     NEGATIVE
           MOVE '*******' TO     P-COMNT
        ELSE
           MOVE SPACES      TO     P-COMNT.
        MOVE     CST-NO      TO     P-NO.
        MOVE     CST-NAME    TO     P-NAME.
        MOVE     CST-BAL     TO     P-BAL.
        IF       LIN-CTR     >      24
                 PERFORM     NEW-PAGE.
        WRITE    P-REC  FROM   DATA-LINE.
        ADD      1      TO     LIN-CTR.
        READ OTMRG AT END
                 MOVE  1    TO    FLAG.
 EXT-MOVE-WRITE.
        EXIT.
 NEW-PAGE.
        MOVE   SPACES     TO        P-REC.
        WRITE  P-REC      AFTER     PAGE.
        MOVE   ZERO       TO        LIN-CTR.
        ADD    1          TO        PAG-CTR.
        MOVE   PAG-CTR    TO        P-PAGE.
        WRITE  P-REC      FROM      HDR1
                              AFTER 1 LINE.
        WRITE  P-REC      FROM      HDR2
                              AFTER 1 LINE.
        WRITE  P-REC      FROM      HDR3
                              AFTER 2 LINE.
 EXTP.
        EXIT.
*

*
*            حل تمرين رقم ٣   في الفصل ٧-٤
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT13.
 AUTHOR.      KHAYAT-JNAID.
*
* يقوم هذا البرنامج بفرز ملف الإدخال *
* وكتابة الملف المفروز على ملف آخر *
*    ثم معالجته.
```

```
*
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.   IBM-370.
   OBJECT-COMPUTER.   IBM-370.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
       SELECT STOCKF ASSIGN  TO UT-S-INF1.
       SELECT SRTFIL ASSIGN  TO UT-S-INF3.
       SELECT OUTSRT ASSIGN  TO UT-S-MRGF.
       SELECT PRINTF ASSIGN  TO UT-S-PRFL.

 DATA DIVISION.
 FILE  SECTION.
*
 FD     STOCKF LABEL RECORD IS STANDARD
            BLOCK CONTAINS 0 RECORDS
                     DATA RECORD IS STK-REC.
 01     STK-REC            PIC  X(80).

* لاحظ ان ملف الفرز عرف ب SD بدل FD: *
 SD     SRTFIL
             RECORD  CONTAINS 80 CHARACTERS
                       DATA RECORD IS SRT-REC.
* لاحظ انه لاضرورة لتوصيف كل الحقول بل *
*                        يكتفي بمفاتيح الفرز
 01     SRT-REC.
  03 SRT-NO         PIC    9(5).
  03 SRT-DATA       PIC    X(75).

*
 FD     OUTSRT LABEL RECORD  STANDARD
           RECORD  CONTAINS 80 CHARACTERS
                       DATA RECORD IS OUT-REC.
 01     OUT-REC.
  03 OUT-NO         PIC    9(5).
  03 OUT-DESC       PIC    X(20).
  03 OUT-QTY        PIC    9(5).
  03 OUT-CODE       PIC    9(3).
  03 OUT-PRC        PIC    9(3)V99.
  03 OUT-MIN        PIC    9(5).
  03 FILLER         PIC    X(37).
```

```
*
 FD     PRINTF    LABEL RECORD  IS OMITTED
                          DATA RECORD IS  P-REC.
 01     P-REC           PIC     X(132).
*
*
 WORKING-STORAGE SECTION.
 01     HDR1.
  03    FILLER  PIC     X(56)   VALUE SPACES.
  03    FILLER  PIC     X(18)   VALUE
                        'S T O C K  L I S T'.
  03    FILLER  PIC     X(56)   VALUE SPACES.
*
 01     HDR2.
  03    FILLER  PIC     X(08)   VALUE
                                  ' DATE :'.
  03    P-DATE  PIC     99/99/99.
  03    FILLER  PIC     X(100)  VALUE SPACES.
  03    FILLER  PIC     X(07)   VALUE
                                   'PAGE :'.
  03    P-PAGE  PIC     ZZ9.
  03    FILLER  PIC     X(5)    VALUE SPACES.
*
 01     HDR3.
  03    FILLER PIC      X(40)   VALUE
      '     ITEM NO.      ITEM DESCRIPTION'.
  03    FILLER PIC      X(29)   VALUE
            '      QUANTITY     ITEM CODE'.
  03    FILLER PIC      X(17)   VALUE
                        '       ITEM PRICE'.
  03    FILLER PIC      X(46)   VALUE
      '     MINIMUM LEVLEL        COMMENTS'.
*
 01     DATA-LINE.
  03    FILLER  PIC     X(06)   VALUE SPACES.
  03    P-NO    PIC     ZZZZ99.
  03    FILLER  PIC     X(06)   VALUE SPACES.
  03    P-DESC  PIC     X(20).
  03    FILLER  PIC     X(09)   VALUE SPACES.
  03    P-QNTY  PIC     ZZZZ9.
  03    FILLER  PIC     X(09)   VALUE SPACES.
```

```
    03     P_CODE   PIC      Z99.
    03     FILLER   PIC      X(10)    VALUE SPACES.
    03     P_PRCE   PIC      ZZ9.99.
    03     FILLER   PIC      X(10)    VALUE SPACES.
    03     P_MINI   PIC      Z(4)9.
    03     FILLER   PIC      X(17)    VALUE SPACES.
    03     P_COMNT  PIC      X(7).
    03     FILLER   PIC      X(05)    VALUE SPACES.
*
*             حقل عمل لقبول التاريخ
*
 01     C_DATE   PIC      9(6).
 01     PAG_CTR  PIC      999      VALUE ZERO.
 01     LIN_CTR  PIC      99       VALUE 25.
 77     FLAG     PIC      9        VALUE ZERO.
*
 PROCEDURE DIVISION.
*    تعليمة فرز ملف الإدخال حسب رقم الصنف :
*    لاحظ ان الملفات لم تفتح بتعليمة OPEN.
*
        SORT    SRTFIL
                ASCENDING KEY SRT_NO
        USING   STOCKF
                GIVING    OUTSRT.
*  بعد تنفيذ تعليمة الفرز السابقة  *
*  تبدأ معالجة الملف OUTSRT معالجة  *
*                     عادية.
*
 START_PROCESSING.
      OPEN INPUT OUTSRT
           OUTPUT   PRINTF.
      ACCEPT C_DATE FROM DATE.
      MOVE C_DATE TO P_DATE.
        READ OUTSRT AT END
                MOVE  1   TO   FLAG.
*
 MAIN_LOOP.
        PERFORM MOVE_WRITE
                    THRU EXT_MOVE_WRITE
         UNTIL  FLAG EQUAL 1.
        CLOSE OUTSRT PRINTF.
        STOP RUN.
```

```
*
 MOVE-WRITE.
          IF        OUT-QTY   NOT     > OUT-MIN
       MOVE     'REORDER'   TO      P-COMNT
         ELSE
       MOVE      SPACES      TO      P-COMNT.
       MOVE     OUT-NO        TO      P-NO.
       MOVE     OUT-DESC      TO      P-DESC.
       MOVE     OUT-QTY       TO      P-QNTY.
       MOVE     OUT-CODE      TO      P-CODE.
       MOVE     OUT-PRC       TO      P-PRCE.
       MOVE     OUT-MIN       TO      P-MINI.
       IF       LIN-CTR       >       24
                PERFORM     NEW-PAGE.
       WRITE   P-REC    FROM    DATA-LINE.
       ADD      1        TO      LIN-CTR.
       READ OUTSRT    AT END
                  MOVE   1    TO    FLAG.
 EXT-MOVE-WRITE.
       EXIT.
 NEW-PAGE.
       MOVE   SPACES     TO        P-REC.
       WRITE P-REC       AFTER     PAGE.
       MOVE   ZERO       TO        LIN-CTR.
       ADD    1          TO        PAG-CTR.
       MOVE   PAG-CTR    TO        P-PAGE.
       WRITE P-REC       FROM      HDR1
                              AFTER 1 LINE.
       WRITE P-REC       FROM      HDR2
                              AFTER 1 LINE.
       WRITE P-REC       FROM      HDR3
                              AFTER 2 LINE.
 EXTP.
       EXIT.
*
```

أنظر شكل المخرجات صفحة ٦٠٠

● حل التمرين رقم ١
الفصل ٨-٦

```
IDENTIFICATION DIVISION.
PROGRAM-ID.     TXT32.
AUTHOR.         KHAYAT-JNAID.
*
ENVIRONMENT     DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. TI990.
OBJECT-COMPUTER. TI990.
*
INPUT-OUTPUT  SECTION.
FILE-CONTROL.
SELECT    INFIL1 ASSIGN TO INPUT "F1".
SELECT    INFIL2 ASSIGN TO INPUT "F2".
SELECT    OUTMRG ASSIGN TO OUTPUT "MG".
*
DATA             DIVISION.
FILE             SECTION.
FD  INFIL1       LABEL RECORD STANDARD.
01  INREC1.
    03 FRST-NO     PIC  9(5).
    03 FRST-DATA   PIC  X(35).
*
FD  INFIL2       LABEL RECORD STANDARD.
01  INREC2.
    03 SCND-NO     PIC 9(5).
    03 SCND-DATA   PIC X(35).
*
FD  OUTMRG      LABEL RECORD STANDARD.
01  MRGREC.
    03 MRG-NO      PIC 9(5).
    03 MRG-DATA    PIC X(35).
*
PROCEDURE    DIVISION.
STARTT.
        OPEN INPUT  INFIL1 INFIL2
             OUTPUT OUTMRG.
*
* قراءه سجل من كل من ملفى الادخال
*
        PERFORM READF1.
        PERFORM READF2.
*
* تكرار الفقره الاساسيه فى البرنامج الى ان
*                 تنتهى سجلات ملفى الادخال
*
        PERFORM COMPAR UNTIL FRST-NO = SCND-NO
                      AND    FRST-NO = HIGH-VALUE.
        CLOSE INFIL1 INFIL2 OUTMRG.
        STOP  RUN.
*
*   مقارنه رقمى السجلين ونقل الاصغر الى ملف الاخراج
* ثم كتابته على ملف الاخراج هذا، بعدها تتم القراءه
*              من الملف المنقول سجله = الاصغر=
```

```
*
 COMPAR.
*
      IF FRST-NO  LESS THAN SCND-NO
                  PERFORM MOVE-1 THRU EXIT1
      ELSE        PERFORM MOVE-2 THRU EXIT2.
*تحريك الحقول من الملف الأول لملف الإخراج
*وقراءة سجل جديد
*
 MOVE-1.
      WRITE MRGREC FROM INREC1.
*
 READF1.
      READ INFIL1 AT END MOVE HIGH-VALUE TO FRST-NO
 EXIT1.
      EXIT.
*تحريك الحقول من الملف الثاني لملف الإخراج
*وقراءة سجل جديد
*
 MOVE-2.
      WRITE MRGREC FROM INREC2.
*
 READF2.
      READ INFIL2 AT END MOVE HIGH-VALUE TO SCND-NO.
 EXIT2.
      EXIT.
```

```
*          حل تمرين رقم ١ الفصل ١٣–٥
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT03.
 AUTHOR.       KHAYAT-JNAID.
*
* يبين هذا التمرين كيفية استخدام
*       PERFORM ...TIMES    تعليمة
*
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.    IBM-370.
   OBJECT-COMPUTER.    IBM-370.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
       SELECT PRINTF  ASSIGN  TO UT-S-PRFL.
```

```
 DATA DIVISION.
 FILE  SECTION.
 FD    PRINTF    LABEL RECORD  IS OMITTED
                         DATA RECORD IS P_REC.
 01    P_REC        PIC     X(132).
*
 WORKING_STORAGE SECTION.
 77    P_CNTR       PIC 99.
 PROCEDURE DIVISION.
       OPEN  OUTPUT PRINTF.
*
       PERFORM PAR_1 5 TIMES.
       CLOSE PRINTF.
       STOP RUN.
*
 PAR_1.
       MOVE     ZERO    TO P_CNTR.
       PERFORM PAR_2  10 TIMES.
       MOVE     SPACES TO P_REC.
       WRITE    P_REC.
 EXT1.
       EXIT.
*
 PAR_2.
       ADD      1       TO P_CNTR.
       MOVE     P_CNTR TO P_REC.
       WRITE    P_REC.
 EXT2.
       EXIT.
```

● حل التمرين رقم ٢ الفصل ١٣ــه

```
IDENTIFICATION DIVISION.
PROGRAM-ID.      CUSTOM.
AUTHOR.          KHAYAT-JNAID.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  TI990.
OBJECT-COMPUTER.  TI990.
SPECIAL-NAMES.    DECIMAL-POINT IS COMMA.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
       SELECT DEPFIL ASSIGN TO INPUT "DF".
       SELECT PRINTF ASSIGN TO PRINT "PF".
```

```
 DATA DIVISION.
 FILE SECTION.
 FD  DEPFIL LABEL RECORDS ARE STANDARD.
  01 CUSTREC.
       03   D-CODE      PIC X.
       03   D-ACNO      PIC 9(5).
       03   D-ACNAME    PIC X(30).
       03   D-ENDBAL    PIC 9(7)V99.
       03   D-DPST      PIC 9(7)V99.
       03   D-DRFT      PIC 9(7)V99.
       03   FILLER      PIC X(17).
*
 FD  PRINTF LABEL RECORD IS OMITTED.
  01 P-REC    PIC X(132).
*
 WORKING-STORAGE SECTION.
 01 HD1.
       03 FILLER PIC X(57) VALUE SPACES.
       03 FILLER PIC X(15) VALUE "COMMERCIAL BANK".
       03 FILLER PIC X(60) VALUE SPACES.
*
 01 HD2.
       03 FILLER PIC X(11) VALUE " DATE : ".
       03 PDY    PIC 99.
       03 FILLER PIC X VALUE "/".
       03 PMT    PIC 99.
       03 FILLER PIC XXX VALUE "/19".
       03 PYR    PIC 99.
       03 FILLER PIC X(31) VALUE SPACES.
       03 FILLER PIC X(23) VALUE
          "REPORT FOR NEW BALANCE".
       03 FILLER PIC X(34) VALUE SPACES.
       03 FILLER PIC X(6)  VALUE "PAGE: ".
       03 PPAGE  PIC ZZ9.
       03 FILLER PIC X(11) VALUE SPACES.
*
 01 HD3.
       03 FILLER PIC X(51) VALUE
          "  ACCOUNT NO       ACCOUNT  NAME ".
       03 FILLER PIC X(42) VALUE
         "LAST BALANCE            DRAFT AMOUNT".
       03 FILLER PIC X(14) VALUE
          "DEPOSIT AMOUNT".
       03 FILLER PIC X(25) VALUE "NEW BALANCE".
*
 01 DATAL.
       03 FILLER PIC X(05) VALUE SPACES.
       03 P-ACNO PIC 9(5).
       03 FILLER PIC X(04) VALUE SPACES.
       03 P-ACNAME PIC X(30).
       03 FILLER PIC X(04) VALUE SPACES.
       03 P-ENDBAL PIC Z(5)99,99.
       03 FILLER PIC X(15) VALUE SPACES.
       03 P-DRFT PIC Z(5)99,99.
       03 FILLER PIC X(07) VALUE SPACES.
```

```
       03 P-DPST PIC Z(5)99,99.
       03 FILLER PIC X(10) VALUE SPACES.
       03 P-NBAL PIC +++++99,99.
       03 FILLER PIC X(11) VALUE SPACES.
*
 01 TOTL1.
       03 FILLER PIC X(46) VALUE SPACES.
       03 FILLER PIC X(27) VALUE
          "TOTAL OF ACTIVE RECORDS :".
       03 T-ACTV PIC ZZZ9.
       03 FILLER PIC X(55) VALUE SPACE.
*
 01 TOTL2.
       03 FILLER PIC X(46) VALUE SPACES.
       03 FILLER PIC X(27) VALUE
          "TOTAL OF LAST BALANCE :".
       03 T-ENDBAL PIC Z(6)99,99.
       03 FILLER PIC X(48) VALUE SPACES.
*
 01 TOTL3.
       03 FILLER PIC X(46) VALUE SPACES.
       03 FILLER PIC X(27) VALUE
          "TOTAL OF DRAFT AMOUNTS : ".
       03 T-DRFT PIC Z(6)99,99.
       03 FILLER PIC X(55) VALUE SPACE.
*
 01 TOTL4.
       03 FILLER PIC X(46) VALUE SPACES.
       03 FILLER PIC X(27) VALUE
          "TOTAL OF DEPOSIT AMOUNTS: ".
       03 T-DPST    PIC Z(6)99,99.
       03 FILLER PIC X(48) VALUE SPACES.
 01 TOTL5.
       03 FILLER PIC X(46) VALUE SPACES.
       03 FILLER PIC X(27) VALUE
          "TOTAL OF NEW BALANCES :".
       03 T-NBAL PIC +++++99,99.
       03 FILLER PIC X(47).
*
* عدادات لتجميع المبالغ النهائية *
*
 77    C-NBAL    PIC  S9(8)V99 VALUE ZEROS.
 77    C-ACTV    PIC   9(4)    VALUE ZEROS.
 77    C-ENDBAL  PIC   9(8)V99 VALUE ZEROS.
 77    C-DPST    PIC   9(8)V99 VALUE ZEROS.
 77    C-DRFT    PIC   9(8)V99 VALUE ZEROS.
*
*  عدادات لرقم الصفحة والسطر
*
 77    LN-CNTR   PIC  99        VALUE ZERO.
 77    PG-CNTR   PIC  999       VALUE ZERO.
```

```
*
*  مساحات عمل لقبول التاريخ من الحاسب  *
*
 01   WDATE.
      03    WYR PIC    99.
      03    WMT PIC    99.
      03    WDY PIC    99.
*
*  مساحة عمل من أجل حساب الرصيد  *
*
 77   D-NBAL     PIC  S9(7)V99.
*
 PROCEDURE      DIVISION.
 OPEN-FILES.
      OPEN         INPUT      DEPFIL
                   OUTPUT     PRINTF.
*
*  قبول التاريخ وتحريك اليوم والشهر والسنة إلى  *
*  إلى سطر العناوين
*
     ACCEPT WDATE FROM DATE.
     MOVE    WYR    TO PYR.
     MOVE    WMT    TO PMT.
     MOVE    WDY    TO PDY.
     PERFORM N-PAGE THRU  E-PAGE.
*
*  قراءة سجل من الملف ثم المعالجة  *
*
READ-A-FILE.
     READ     DEPFIL AT END GO TO END-OF-JOB.
*  اختبار فيما إذا السجل مقبل أم لا ؟  *
     IF       D-CODE = "D" GO TO READ-A-FILE.
*
*        حساب الرصيد الجديد
*
     COMPUTE D-NBAL ROUNDED =
                         D-ENDBAL + D-DPST - D-DRFT.
*
*تحريك الحقول من السجل الإدخال إلى سجل الطباعة
*  وكذلك حقل الرصيد الجديد المحسوب
*
         MOVE      D-ACNO        TO       P-ACNO.
         MOVE      D-ACNAME      TO       P-ACNAME.
         MOVE      D-ENDBAL      TO       P-ENDBAL.
         MOVE      D-DRFT        TO       P-DRFT.
         MOVE      D-DPST        TO       P-DPST.
         MOVE      D-NBAL        TO       P-NBAL.
*
*  اختبار نهاية الصفحة  *
*
         IF        LN-CNTR       >        22
                   PERFORM    N-PAGE THRU E-PAGE.
```

```
*
* طباعة سجل على ملف الطابعة
*
        WRITE P-REC          FROM  DATAL
                             AFTER    2.
        ADD   1              TO    LN-CNTR.
*
* تجميع القيم المطلوب طباعتها فى نهاية التقرير
*
        ADD   1              TO    C-ACTV.
        ADD   D-ENDBAL       TO    C-ENDBAL.
        ADD   D-DPST         TO    C-DPST.
        ADD   D-DRFT         TO    C-DRFT.
        ADD   D-NBAL         TO    C-NBAL.
* بعد الانتهاء من معالجة السجل
* نذهب لمعالجة سجل آخر
*
        GO    TO             READ-A-FILE.
*
* الفقره التاليه هى لقلب الصفحة وطباعة العناوين
*
N-PAGE.
        MOVE   ZERO         TO LN-CNTR.
        ADD    1            TO PG-CNTR.
        MOVE   PG-CNTR      TO PPAGE.
        MOVE   SPACES       TO P-REC.
        WRITE  P-REC        FROM HD1 AFTER PAGE.
        MOVE   SPACES       TO P-REC.
        WRITE  P-REC        FROM HD2 AFTER 2 LINES.
        MOVE   SPACES       TO P-REC.
        WRITE  P-REC        FROM HD3 AFTER 3 LINES.
        MOVE   SPACES       TO P-REC.
E-PAGE.
        EXIT.
*
* الفقره التاليه هى لطباعة المجاميع النهائيه
*
END-OF-JOB.
        MOVE   C-ACTV       TO T-ACTV.
        MOVE   C-ENDBAL     TO T-ENDBAL.
        MOVE   C-DRFT       TO T-DRFT.
        MOVE   C-DPST       TO T-DPST.
        MOVE   C-NBAL       TO T-NBAL.
*
        WRITE  P-REC        FROM SPACE AFTER PAGE.
        WRITE  P-REC        FROM TOTL1 AFTER 32.
        WRITE  P-REC        FROM TOTL2 AFTER 2.
        WRITE  P-REC        FROM TOTL3 AFTER 2.
        WRITE  P-REC        FROM TOTL4 AFTER 2.
        WRITE  P-REC        FROM TOTL5 AFTER 2.
*
* نهاية العمل وإغلاق الملفات
*
        CLOSE DEPFIL, PRINTF.
        STOP  RUN.
```

أنظر شكل المخرجات صفحة ٦٠١.

```
*           حل تمرين رقم ٦ للفصل ١٣-٥
 IDENTIFICATION DIVISION.
 PROGRAM-ID.  TXT19.
 AUTHOR.        KHAYAT-JNAID.
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER.  IBM-370.
 OBJECT-COMPUTER.  IBM-370.
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
      SELECT VALFILE ASSIGN TO UT-S-SYSIN.
      SELECT PRINTFL ASSIGN TO UT-S-PRF.
*
 DATA DIVISION.
 FILE SECTION.
 FD   VALFILE RECORD CONTAINS 80 CHARACTERS,
                BLOCK CONTAINS 0 RECORDS
                      LABEL RECORD IS STANDARD.
 01   VAL-REC.
      03 VAL-NO          PIC 9(5).
      03 VAL-NAME        PIC X(20).
      03 VAL-ADRESS      PIC X(24).
      03 VAL-DATE        PIC 9(8).
      03 VAL-CODE        PIC 99.
      03 VAL-AREA        PIC 99.
      03 VAL-BLNK        PIC X(19).
*
 FD   PRINTFL RECORD CONTAINS 132 CHARACTERS
                     LABEL RECORD IS OMITTED.
 01   P-REC                PIC X(132).
*
 WORKING-STORAGE SECTION.
 01   HDR1.
      03 FILLER PIC X(29) VALUE SPACES.
      03 FILLER PIC X(8)  VALUE '| CODE |'.
      03 FILLER PIC X(16) VALUE
                      '     AREA NAME  |'.
      03 FILLER PIC X(20) VALUE
                     '  SUBSCRIBER NO.   |'.
      03 FILLER PIC X(60) VALUE SPACES.
```

```
*
 01    DLN.
       03 FILLER PIC X(29) VALUE SPACES.
       03 FILLER PIC X VALUE '|'.
       03 FILLER PIC XX VALUE SPACES.
       03 P-CODE PIC Z9.
       03 FILLER PIC XX VALUE SPACES.
       03 FILLER PIC X  VALUE '|'.
       03 FILLER PIC XX VALUE SPACES.
       03 P-AREA PIC X(10).
       03 FILLER PIC XXX VALUE SPACES.
       03 FILLER PIC X  VALUE '|'.
       03 FILLER PIC X(9) VALUE SPACES.
       03 P-CNTR PIC ZZ9.
       03 FILLER PIC X(4) VALUE SPACES.
       03 FILLER PIC X(8) VALUE '    |    '.
       03 FILLER PIC X(50) VALUE SPACES.
*
 01    DASHES.
       03 FILLER PIC X(29) VALUE SPACES.
       03 FILLER PIC X(44) VALUE ALL '-'.
       03 FILLER PIC X(30) VALUE SPACES.
*
 01    TOTL.
       03 FILLER PIC X(29) VALUE SPACES.
       03 FILLER PIC X VALUE '|'.
       03 FILLER PIC X(14) VALUE SPACES.
       03 FILLER PIC X(17) VALUE ' OVERALL TOTAL:'.
       03 POVALL PIC ZZ99.
       03 FILLER PIC X(04) VALUE SPACES.
       03 FILLER PIC X(8) VALUE '    |    '.
       03 FILLER PIC X(50) VALUE SPACES.
 01    TBLE-AREA.
       03 FILLER PIC X(15) VALUE
                           '01RIYADH     000'.
       03 FILLER PIC X(15) VALUE
                           '02JEDDAH     000'.
       03 FILLER PIC X(15) VALUE
                           '03DAMMAM     000'.
       03 FILLER PIC X(15) VALUE
                           '04QUASEEM    000'.
```

```
       03 FILLER PIC X(15) VALUE
                           '05HAYEL     000'.
       03 FILLER PIC X(15) VALUE
                           '06ALMADINA  000'.
       03 FILLER PIC X(15) VALUE
                           '07ABHA      000'.
       03 FILLER PIC X(15) VALUE
                           '08BALGURASHI000'.
       03 FILLER PIC X(15) VALUE
                           '09ASEER     000'.
       03 FILLER PIC X(15) VALUE
                           '10SHAQRA    000'.
*
 01    RED-TBL-AREA REDEFINES TBLE-AREA.
       02 TBL-ITM OCCURS 10 TIMES ASCENDING
          KEY TBL-CODE INDEXED BY IX.
         03  TBL-CODE   PIC  99.
         03  TBL-AREA   PIC X(10).
         03  TBL-CNTR   PIC 999.
*
 77  OVALL-CNTR        PIC 9(4) VALUE ZERO.
 PROCEDURE DIVISION.
 STRT.
     OPEN INPUT VALFILE OUTPUT PRINTFL.
 RD-RC.
     READ VALFILE AT END GO TO END-PROCESS.
     SEARCH  ALL TBL-ITM  AT END GO TO RD-RC
            WHEN VAL-AREA = TBL-CODE(IX)
                       ADD 1 TO TBL-CNTR(IX).
     ADD 1 TO OVALL-CNTR.
     GO TO RD-RC.
*
 END-PROCESS.
     WRITE P-REC AFTER PAGE.
     WRITE P-REC FROM DASHES AFTER 10.
     MOVE SPACES TO P-REC.
     WRITE P-REC FROM HDR1.
     MOVE SPACES TO P-REC.
     WRITE P-REC FROM DASHES.
     PERFORM LAST-PRINT THRU EXTP
               VARYING IX FROM 1 BY 1
                           UNTIL IX > 10.
```

```
    MOVE OVALL-CNTR TO POVALL.
    WRITE P-REC FROM TOTL.
    WRITE P-REC FROM DASHES.
    CLOSE VALFILE PRINTFL.
    STOP RUN.
LAST-PRINT.
    MOVE TBL-CODE(IX) TO P-CODE.
    MOVE TBL-AREA(IX) TO P-AREA.
    MOVE TBL-CNTR(IX) TO P-CNTR.
    MOVE SPACES TO P-REC.
    WRITE P-REC FROM DLN AFTER 1.
    MOVE SPACES TO P-REC.
    WRITE P-REC FROM DASHES AFTER 1.
EXTP.
    EXIT.
```

شكل المخرجات

CODE	AREA NAME	SUBSCRIBER NO.
1	RIYADH	5
2	JEDDAH	2
3	DAMMAM	4
4	QUASEEM	3
5	HAYEL	3
6	ALMADINA	4
7	ABHA	2
8	BALGURASHI	3
9	ASEER	0
10	SHAQRA	3
	OVERALL TOTAL :	29

```
*     ١٣-٥ حل تمرين رقم ٣   فى الفصل                        *
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT15.
 AUTHOR.       KHAYAT-JNAID.
*
*     يقوم هذا البرنامج بمعالجة ملف مفهرس                        *
*
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.    IBM-370.
   OBJECT-COMPUTER.    IBM-370.
   SPECIAL-NAMES.      DECIMAL-POINT IS COMMA.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
        SELECT PAYMST  ASSIGN  TO PAYF
        ORGANIZATION   IS INDEXED;
        ACCESS MODE    SEQUENTIAL;
        RECORD KEY     EMP-NO.
        SELECT P-FILE  ASSIGN  TO UT-S-PRFL.
 DATA DIVISION.
 FILE  SECTION.
 FD     PAYMST     LABEL RECORD  IS STANDARD
                   DATA RECORD IS PAY-REC.
 01     PAY-REC.
        03     EMP-NO     PIC  X(5).
        03     EMP-NAME   PIC  X(30).
        03     EMP-SAL    PIC  9(5)V99.
        03     EMP-HRS1   PIC  9(3).
        03     EMP-HRS2   PIC  9(3).
        03     EMP-HRS3   PIC  9(3).
        03     EMP-OVTM   PIC  9(5)V99.
        03     FILLER    PIC  X(22).
*
 FD     P-FILE     LABEL RECORD  IS OMITTED
                   DATA RECORD IS  P-REC.
 01     P-REC          PIC     X(132).
*
 WORKING-STORAGE SECTION.
 01     HDR-1.
  03    FILLER  PIC     X(56)   VALUE SPACES.
  03    FILLER  PIC     X(19)   VALUE
```

```
                                  .'شــــــــركة الـلـيـــــــن'
  03    FILLER   PIC     X(57)    VALUE SPACES.
 01     HDR-2.
  03    FILLER   PIC     X(20)    VALUE
                                 '    رقم الموظف       '.
  03    FILLER   PIC     X(30)    VALUE
                                 '   الإســــــــم      '.
  03    FILLER   PIC     X(30)    VALUE
                  '   الراتب              .'خارج الدوام
  03    FILLER   PIC     X(52)    VALUE
                                        '   .'ملاحظات
*
 01     DATA-LINE.
  03    FILLER      PIC      X(02)     VALUE SPACES.
  03    P-NO        PIC      ZZZ99.
  03    FILLER      PIC      X(08)     VALUE SPACES.
  03    P-NAME      PIC      X(30).
  03    FILLER      PIC      X(04)     VALUE SPACES.
  03    P-SAL       PIC      Z(4)9,99.
  03    FILLER      PIC      X(06)     VALUE SPACES.
  03    P-OVTM      PIC      ZZZ99,99.
  03    FILLER      PIC      X(09)     VALUE SPACES.
  03    P-COMN      PIC      X(16).
  03    FILLER      PIC      X(10)     VALUE SPACES.

*
 01     TOTAL-L1.
  03    FILLER      PIC      X(39)     VALUE SPACES.
  03    PCNTR1      PIC      Z(5)9,99.
  03    FILLER      PIC      X(84)     VALUE
                             ' :  مجموع الإجـــــــــور'.
 01     TOTAL-L2.
  03    FILLER      PIC      X(39)     VALUE SPACES.
  03    PCNTR2      PIC      Z(5)9,99.
  03    FILLER      PIC      X(84)     VALUE
                             ' : مجموع خارج الدوام'.
 01     TOTAL-L3.
  03    FILLER      PIC      X(39)     VALUE SPACES.
  03    PCNTR3      PIC      Z(5)9,99.
  03    FILLER      PIC      X(84)     VALUE
                             ' :  عدد العمال الكلي'.
```

```
 *
 *      عدادات تستخدم للتجميع
  01      COUNTERS.
          03 CNTR1 PIC      9(6)V99.
          03 CNTR2 PIC      9(6)V99.
          03 CNTR3 PIC      9999.
  77      HALF-SAL PIC      9(5)V99.
  77     HOURS-RAT PIC      9(3)V99.
  77     E-O-J      PIC      XXX      VALUE 'NO'.
  77     PEMK       PIC      X(16)    VALUE
                              'تجاوز الحد الاعلى'.
 *
 PROCEDURE DIVISION.
        OPEN        I-O      PAYMST
                    OUTPUT   P-FILE.
 *      اعطاء القيمة صفر للعدادات
        MOVE        ZEROS          TO    COUNTERS.
        WRITE    P-REC   AFTER PAGE.
        WRITE    P-REC   FROM  HDR-1.
        WRITE    P-REC   FROM  HDR-2 AFTER 2.
        PERFORM READ-FILE.
 *  الحل الاساسي للبرنامج  *
 MAIN-LOOP.
        PERFORM READ-RTN THRU EXT-RTN
                           UNTIL E-O-J = 'YES'.
        PERFORM FINAL-RTN.
        CLOSE    PAYMST P-FILE.
        STOP     RUN.
  READ-RTN.
 *
 *  :  حساب اجرة الساعة
        COMPUTE HOURS-RAT  ROUNDED =
                           EMP-SAL / ( 30 * 8 ).
 *  :  حساب خارج الدوام
        COMPUTE EMP-OVTM   ROUNDED =
              HOURS-RAT * ( EMP-HRS1 +
                            EMP-HRS2 * 1,5 +
                            EMP-HRS3 * 2      ).
 *  :  حساب نصف الدوام
        DIVIDE EMP-SAL BY 2 GIVING HALF-SAL.
```

```
*
*  :   مقارنه نصف الراتب مع خارج الدوام
       IF EMP-OVTM  GREATER THAN HALF-SAL
                   MOVE HALF-SAL TO EMP-OVTM.
       PERFORM       PRINT-LINE.
       ADD       EMP-SAL              TO CNTR1.
       ADD       EMP-OVTM             TO CNTR2.
       ADD       1                    TO CNTR3.
*            اعاده كتابه السجل على نفس الملف
*         المفهرس بعد تعديل حقل خارج الدوام
       REWRITE PAY-REC.
* : قراءه سجل جديد من الملف المفهرس
*
 READ-FILE.
       READ      PAYMST NEXT
                 AT END MOVE 'YES' TO E-O-J.
*
  EXT-RTN.
       EXIT.
*
  PRINT-LINE.
*  : تحريك الحقول والنتائج الى سطر الطباعه
       MOVE      EMP-NO       TO      P-NO.
       MOVE      EMP-NAME     TO      P-NAME.
       MOVE      EMP-SAL      TO      P-SAL.
       MOVE      EMP-OVTM     TO      P-OVTM.
*
*  :   كتابه السجل على ملف الطباعه
       WRITE     P-REC        FROM
          DATA-LINE  AFTER ADVANCING 2 LINES.
 EXT-P.
        EXIT.
*
* :      طباعه اسطر المجاميع النهائيه
 FINAL-RTN.
       MOVE      CNTR1    TO     PCNTR1.
       WRITE     P-REC    FROM TOTAL-L1.
       MOVE      CNTR2    TO     PCNTR2.
       WRITE     P-REC    FROM TOTAL-L2.
       MOVE      CNTR3    TO     PCNTR3.
       WRITE     P-REC    FROM TOTAL-L3.
 EXITF.
       EXIT.
```

```
*           حل تمرين رقم ٥    في الفصل ١٣ـن
 IDENTIFICATION DIVISION.
 PROGRAM-ID. TXT17.
 AUTHOR.       KHAYAT-JNAID.
*
*    يقوم هذا البرنامج بعمل تحقق لملف تعديل
*  مع الملف الاساسي ويطبع السجل الذى ليس له اصل
*
 ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
   SOURCE-COMPUTER.    IBM-370.
   OBJECT-COMPUTER.    IBM-370.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
        SELECT SUBFILE ASSIGN  TO UT-S-SFIL.
        SELECT INNFILE ASSIGN  TO UT-S-INFL.
        SELECT CONFILE ASSIGN  TO UT-S-CONF.
        SELECT PRINTFL ASSIGN  TO UT-S-PFLL.
 DATA DIVISION.
 FILE  SECTION.
*
*  :  توصيف الملف الاساسي
 FD     SUBFILE   LABEL RECORD  IS STANDARD
            BLOCK CONTAINS 0 RECORDS
                     DATA RECORD IS SUB-REC.
 01     SUB-REC.
        03  SUB-NO       PIC  X(5).
        03  SUB-NAME     PIC  X(20).
        03  SUB-ADRESS   PIC  X(24).
        03  SUB-DATE.
          05    SUB-YEAR PIC  9(4).
          05    SUB-MNTH PIC   9(2).
          05    SUB-DAY  PIC  9(2).
        03  SUB-CODE     PIC  99.
        03  SUB-AREA     PIC  99.
        03  SUB-CNTR     PIC  9(7).
        03  SUB-LREAD    PIC  9(7).
        03  FILLER       PIC  X(5).
*
 FD     INNFILE   LABEL RECORD  IS STANDARD
            BLOCK CONTAINS 0 RECORDS
                     DATA RECORD IS IN-REC.
```

```
 01     IN-REC.
        03   IN-NO       PIC  X(5).
        03   IN-READ     PIC  9(7).
        03   IN-DATE     PIC  9(8).
        03  IN-COMNT     PIC  X(30).
        03  FILLER       PIC  X(30).
*
 FD     CONFILE   LABEL RECORD  IS STANDARD
            BLOCK CONTAINS 0 RECORDS
                     DATA RECORD IS CON-REC.
 01    CON-REC.
        03  CON-NO       PIC  9(5).
        03  CON-READ     PIC  9(7).
        03  CON-DATE     PIC  9(8).
        03  CON-COMNT    PIC  X(30).
        03  FILLER       PIC  X(30).
*
* توصيف ملف الطباعة : *
 FD     PRINTFL   LABEL RECORD  IS OMITTED
                     DATA RECORD IS  P-REC.
 01     P-REC            PIC     X(132).
*
 WORKING-STORAGE SECTION.
*
 01     HDR-1.
  03    FILLER    PIC     X(58)    VALUE SPACES.
  03    FILLER    PIC     X(16)    VALUE
                                 'شركة الكهرباء'.
  03    FILLER    PIC     X(58)    VALUE SPACES.
*
 01     HDR-2.
  03    DAY-P     PIC     99.
  03    FILLER    PIC     X        VALUE '/'.
  03    MNTH-P    PIC     99.
  03    FILLER    PIC     XXX      VALUE '/19'.
  03    YEAR-P    PIC     99.
  03    FILLER    PIC     X(12) VALUE
                                 'تاريخ :'.
  03    FILLER    PIC     X(29)    VALUE SPACES.
  03    FILLER    PIC     X(38)    VALUE
     'قائمة بالسجلات المرفوضة من ملف الإستهلاك'.
  03    FILLER    PIC     X(29)    VALUE SPACES.
```

```
   03     P-CTR       PIC     ZZ9.
   03     FILLER      PIC     X(09)     VALUE
*                                          'صفحة رقم: '.
 01       HDR-3.
   03     FILLER      PIC     X(20)     VALUE SPACES.
   03     FILLER      PIC     X(16)     VALUE
                                      '   رقم المشترك  '.
   03     FILLER      PIC     X(16)     VALUE
                                      '   مقدار القراءة  '.
   03     FILLER      PIC     X(07)     VALUE
                                                SPACES.
   03     FILLER      PIC     X(16)     VALUE
                                      '   تاريخ القراءة  '.
   03     FILLER      PIC     X(24)     VALUE
                                                SPACES.
   03     FILLER      PIC     X(10)     VALUE
                                             '  ملاحظات  '.
*
 01       DATA-LINE.
   03     FILLER      PIC     X(20)     VALUE SPACES.
   03     FILLER      PIC     X(04)     VALUE SPACES.
   03     P-NO        PIC     XXXXX.
   03     FILLER      PIC     X(07)     VALUE SPACES.
   03     P-READ      PIC     Z(4)999.
   03     FILLER      PIC     X(19)     VALUE SPACES.
   03     P-DATE      PIC     9999/99/99.
   03     FILLER      PIC     X(26)     VALUE SPACES.
   03     P-COMNT     PIC     X(30).
*
*
* لقبول التاريخ من الحاسب:
 01       CURENT-DATE.
   03     C-YEAR      PIC     99.
   03     C-MNTH      PIC     99.
   03     C-DAY       PIC     99.
*
* الحقل التالي يستخدم لاختبار نهاية الملف:
 77       EOJOB       PIC     XXX       VALUE SPACES.
*
* عدادات تستخدم للتجميع :
 77       LINE-CTR PIC        99        VALUE 30.
 77       PAGE-CTR PIC        99        VALUE ZERO.
```

```
*
 PROCEDURE DIVISION.
       OPEN       INPUT    SUBFILE INNFILE
                  OUTPUT   PRINTFL CONFILE.
*
*        قبول التاريخ وطباعة العناوين
*        وقراءة السجل الاول من الملف
*
       ACCEPT     CURENT-DATE FROM DATE.
       MOVE       C-YEAR          TO    YEAR-P.
       MOVE       C-MNTH          TO    MNTH-P.
       MOVE       C-DAY           TO    DAY-P.
       PERFORM    NEW-PAGE        THRU EXTP.
*
*  المعالجة الرئيسية للبرنامج
 MAIN-LOOP.
       PERFORM    READ-TRANS   THRU READ-MASTER.
  COMPAR.
*
*
*
        IF EOJOB = 'YES'
                CLOSE SUBFILE INNFILE
                   CONFILE PRINTFL STOP RUN.
        IF IN-NO < SUB-NO PERFORM PRINT-LINE
                          PERFORM READ-TRANS
                          GO TO COMPAR
           ELSE
        IF SUB-NO < IN-NO PERFORM READ-MASTER
                          GO TO COMPAR
           ELSE
        WRITE CON-REC FROM IN-REC.
        GO TO MAIN-LOOP.
* قراءة سجل من ملف التعديل
 READ-TRANS.
        READ INNFILE AT END
                        MOVE   'YES' TO EOJOB.
 EXTT.
       EXIT.
* قراءة سجل من الملف الاساسي
 READ-MASTER.
        READ SUBFILE AT END
```

```
                    MOVE HIGH-VALUE TO SUB-NO.
 EXTM.
      EXIT.
 PRINT-LINE.
        MOVE   IN-NO     TO P-NO.
        MOVE   IN-READ   TO P-READ.
        MOVE   IN-DATE   TO P-DATE.
        MOVE   IN-COMNT  TO P-COMNT.
        IF     LINE-CTR GREATER THAN 29
               PERFORM NEW-PAGE THRU EXTP.
        WRITE P-REC FROM DATA-LINE
                   AFTER ADVANCING 2 LINES.
        ADD 1 TO LINE-CTR.
*       فقرة قلب الصفحة وطبع العناوين :
 NEW-PAGE.
        MOVE   SPACES      TO         P-REC.
        WRITE  P-REC       AFTER      PAGE.
        MOVE   ZERO        TO         LINE-CTR.
        ADD    1           TO         PAGE-CTR.
        MOVE   PAGE-CTR    TO         P-CTR.
        WRITE P-REC  FROM HDR-1 AFTER 1 LINE.
        WRITE P-REC  FROM HDR-2 AFTER 1 LINE.
        WRITE P-REC  FROM HDR-3 AFTER 1 LINE.
 EXTP.
        EXIT.
*
```

أنظر شكل المخرجات صفحة ٦٠١

أشكال مخرجات
ونتائج بعض التمارين

نتيجة البرنامج

LOCATION	AGE
RIYADH	27
RIYADH	29
RIYADH	24
RIYADH	27
RIYADH	28
RIYADH	23
RIYADH	33
RIYADH	32
RIYADH	30
RIYADH	33
RIYADH	33
RIYADH	33
RIYADH	33
RIYADH	33

N A M E	S A L A R Y
ZUHAIR KHAYAT	2500
ALI ALGHAMDI	2000
SALEH MONAJED	3000
ABDULLAH GHANEM	2500
ALI SALEH	3000
FERAS A·MAD	2000
KHALDOON JNAID	2000
SALEM A:DULLA	2500
IBRAHIM ALHARBI	2500
IBRAHIM AOTAIBI	2400
SABER ALYOUSEF	2400
HAYTHAM ALZOUHD	3000
ISA ALMADANI	3000
KHALED NJAIM	3000

قائمة اخطاء تعديل

الملاحظات	اسم
بجل غير موجود	RAJAB
جل غير موجود	AHMED JNAID
جل غير موجود	AHMED
جل غير موجود	KHANAWATI

الملف الاساسي

المشترك	رقم المشترك
SUBHI	10008
FERAS	10017
KHALDOON	10018
SALEM	10023

عدد السجلات المضافه 25

عدد السجلات المعدله 3

عدد السجلات المحذوفه 0

عدد السجلات المرفوضه 4

SUBSCRIPTION DATE	SUBS. CODE	SUBS. AREA	COUNTR	LAST RD
11/12/1404	01	01	1233	100
11/12/1404	01	01	1234	100
11/12/1404	02	01	1235	0
12/01/1404	02	01	1236	200
12/01/1404	02	01	1237	0
12/01/1404	03	01	1238	300
12/01/1404	03	01	1240	400
12/01/1404	01	01	1242	500
02/01/1405	01	01	1243	0
02/01/1405	04	01	1244	700
02/01/1405	04	01	1245	0
02/01/1405	04	01	1200	0
02/01/1405	05	01	1203	0
02/01/1405	01	01	1205	0
02/01/1405	05	01	1206	0
10/01/1405	06	01	1207	0
02/01/1405	02	01	1208	200
02/02/1405	07	01	1209	200
02/02/1405	07	01	1213	100
02/03/1405	06	01	1333	900
02/03/1405	02	01	1343	800
02/03/1405	03	01	1453	700
02/03/1405	03	01	1563	100
02/03/1405	08	01	1673	600
02/04/1405	08	01	983	400
02/04/1405	09	01	1783	200
02/04/1405	01	01	1903	200
02/04/1405	02	01	1913	200
02/04/1405	02	01	1983	200

ELECTRIC COMPANY
LIST OF SUBSCRIBER RECORDS

DATE: 21/11/1984

SUBSCRIBER NO.	SUBS. NAME	SUBS. ADDRESS
10004	ZUHAIR KHAYAT	RYIADH P.O BOX 112233
10005	SAMI ALHALABI	RYIADH P.O BOX 2066
10006	MOJAHED ALHARBI	RYIADH P.O BOX 230
10007	BADR ALJUNDI	RYIADH P.O BOX 1102
10008	SUBHI RAJAB	RYIADH P.O BOX 102
10009	AHMAD HAMEED GHAZI	RYIADH P.O BOX 1002
10017	FERAS AHMED JNAID	RYIADH P.O BOX 110
10018	KHALDOON AHMED	RYIADH P.O BOX 1202
10019	SAMEER HUSNI	RYIADH P.O BOX 1103
10021	MANSUR AL GHADAN	RYIADH P.O BOX 1105
10022	ABDULAZIZ QUAIZ	RYIADH P.O BOX 1023
10023	SALEM KHANAWATI	RYIADH P.O BOX 1103
10024	BASHIR ZUNDAH	RYIADH P.O BOX 210
10027	ALI AL YUSEF	RYIADH P.O BOX 230
10029	MOHAMED ZAKI	RYIADH P.O BOX 103
10033	MOLHAM ZAINI	RYIADH P.O BOX 450
10034	AHMED JAMALI	RYIADH P.O BOX 44
10036	ABDULRAHMAN HARBI	RYIADH P.O BOX 11
10038	ABDUL AZIZ MANSOOR	RYIADH P.O BOX 4567
10039	GHIAS ALCHATTA	RYIADH P.O BOX 234
10048	HUSNI AL SHMEMRI	RYIADH P.O BOX 556
10049	MONEER QUATARI	RYIADH P.O BOX 678
10050	MOHAMED ABD ALAZIZ	RYIADH P.O BOX 45
10051	NASER AL OHIBI	RYIADH P.O BOX 32
10052	OTHMAN AL BASHEER	RYIADH P.O BOX 1443
10053	ALMAJED YAHYAWI	RYIADH P.O BOX 1244
10056	MAAMAR MAKID	RYIADH P.O BOX 1223
10058	LINAH KHAYAT	RYIADH P.O BOX 1026
10059	YOUSUF QABOLI	RYIADH P.O BOX 1003

DATE: 21/11/1984

SOCIAL SECURITY	SOCIAL_FUND	NET PAY	SIGNATURE
250,00	190,00	5160,00	
200,00	140,00	4060,00	
300,00	240,00	5960,00	
200,00	140,00	4060,00	
65,00	19,00	1616,00	
115,00	55,00	2530,00	
220,00	160,00	4420,00	
275,00	215,00	5410,00	
50,00	10,00	1340,00	
50,00	10,00	1340,00	
115,00	55,00	2530,00	
225,00	165,00	4510,00	
195,00	135,00	3970,00	
335,00	275,00	6490,00	
115,00	55,00	2530,00	
45,00	9,00	1246,00	
45,00	9,00	1246,00	
40,00	8,00	1152,00	
170,00	110,00	3520,00	
225,00	165,00	4610,00	
35,00	7,00	1158,00	
225,00	165,00	4610,00	
225,00	165,00	4610,00	
170,00	110,00	3620,00	
100,00	40,00	2360,00	
115,00	55,00	2630,00	
100,00	40,00	2360,00	
75,00	25,00	1700,00	
70,00	22,00	1808,00	

BRIGHT STAR COMPANY
PAYROLL REPORT

PAGE NO : 1

EMPLOYEE NO.	EMPLOYEE NAME	BASE SALARY	TRANSPORTATION
1004	ZUHAIR KHAYAT	5000,00	600,00
1005	SAMI ALHALABI	4000,00	400,00
1006	MOJAHED ALHARBI	6000,00	500,00
1007	BADR ALJUNDI	4000,00	400,00
1008	SUBHI RAJAB	1300,00	400,00
1009	AHMAD HAMEED GHAZI	2300,00	400,00
1017	FERAS AHMED JNAID	4400,00	400,00
1018	KHALDOON AHMED	5500,00	400,00
1019	SAMEER HUSNI	1000,00	400,00
1021	MANSUR AL GHADAN	1000,00	400,00
1022	ABDULAZIZ QUAIZ	2300,00	400,00
1023	SALEM KHANAWATI	4500,00	400,00
1024	BASHIR ZUNDAH	3900,00	400,00
1027	ALI AL YUSEF	6700,00	400,00
1029	MOHAMED ZAKI	2300,00	400,00
1033	MOLHAM ZAINI	900,00	400,00
1034	AHMED JAMALI	900,00	400,00
1036	ABDULRAHMAN HARBI	800,00	400,00
1038	ABDUL AZIZ MANSOOR	3400,00	400,00
1039	GHIAS ALCHATTA	4500,00	500,00
1048	HUSNI AL SHMEMRI	700,00	500,00
1049	MONEER QUATARI	4500,00	500,00
1050	MOHAMED ABD ALAZIZ	4500,00	500,00
1051	NASER AL OHIBI	3400,00	500,00
1052	OTHMAN AL BASHEER	2000,00	500,00
1053	ALMAJED YAHYAWI	2300,00	500,00
1056	MAAMAR MAKID	2000,00	500,00
1058	LINAH KHAYAT	1500,00	500,00
1059	YOUSUF QABOLI	1400,00	500,00

شركة الكهرباء

قائمة بالمشتركين واستهلاكهم
لشهر٢٠ عام ١٤٠٥

رقم : ١

رمز المنطقة	عنوان المشترك	اسم المشترك	المشترك
٠١	RYIADH P,O BOX ١١٢٢٣٣	ZUHAIR KHAYAT	١٠
٠١	RYIADH P,O BOX ٢٠٦٦	SAMI ALHALABI	١٠
٠١	RYIADH P,O BOX ٢٣٠	MOJAHED ALHARBI	١٠
٠١	RYIADH P,O BOX ١١٠٢	BADR ALJUNDI	١٠
٠١	RYIADH P,O BOX ١١٠	FERAS AHMED JNAID	١٠
٠١	RYIADH P,O BOX ١١٠٣	SAMEER HUSNI	١٠
٠١	RYIADH P,O BOX ٤٥٠	MOLHAM ZAINI	١٠
٠١	RYIADH P,O BOX ٣٢	NASER AL OHIBI	١٠
٠١	RYIADH P,O BOX ١٤٤٣	OTHMAN AL BASHEER	١٠
٠١	RYIADH P,O BOX ١٢٤٤	ALMAJED YAHYAWI	١٠
٠١	RYIADH P,O BOX ١٢٢٣	MAAMAR MAKID	١٠
٠١	RYIADH P,O BOX ١٠٢٦	LINAH KHAYAT	١٠
٠١	RYIADH P,O BOX ١٠٠٣	YOUSUF QABOLI	١٠

تاريخ ٢١/١١/١٩٨٤

رمز الاستهلاك	رقم العداد	مقدار الاستهلاك	المستحق للدفع
٠١	١٢٣٣	٢٣٠	١١,٦٤
٠١	١٢٣٤	٥٣٠	١٩,٦٨
٠٢	١٢٣٥	١٥٥٠	٥٦,٤٠
٠٢	١٢٣٦	٣٣٠	١٦,٤٤
٠٣	١٢٤٠	٢٥٤٥	٩٢,٢٢
٠١	١٢٤٣	١٦٧٠	٦٠,٧٢
٠٦	١٢٠٧	١٤٥٥	٠,٦٠
٠٨	١٦٧٣	٤٥٤٦	١٠٩,٧٠
٠٨	٩٨٣	٢٢٢٢	٥٣,٩٢
٠٩	١٧٨٣	١٨٢	٤,٩٦
٠١	١٩٠٣	٧٥٢	٢٧,٦٧
٠٢	١٩١٣	٣٣٢	١٦,٥٣
٠٢	١٩٨٣	٤٥٢	٢٢,٢٩

الخلاصة النهائية

عدد المستهلكين : ٠١٣

مقدار الكيلو واط المصروفة :١٦٧٩٦

مقدار المبالغ المستحقة :٤٩٢,٧٧

PAGE NO : 1

SUBSCRIPTION DATE	SUB. C O D E	SUB. AREA	LAST READING
1405/01/02	01	01	1243
1405/01/02	05	01	1203
1005/00/00	05	01	1203
* * *			
1405/01/02	05	00	61206
		*	
0000/00/00	00	00	00
* * *	*	*	*

PAGE :

ITEM PRICE	MINIMUM LEVLEL	COMMENTS
22.00	23	REORDER
100.00	25	REORDER
32.00	100	
2.00	100	
10.00	400	REORDER
3.00	100	
10.00	50	
1.00	200	REORDER
3.00	300	REORDER
2.00	100	
22.00	23	REORDER

```
                         ELECTRIC COMPANY

DATE: 21/11/1984       LIST OF INVALID SUBSCRIBER RECORDS

SUBSCRIBER NO.     SUBS. N A M E        SUBS. A D D R E S S

   1DD19        AYMAN KHALIFA          RYIADH P.O BOX 4103
    *
   10025        YAQUB JAMEEL
                                                *
   10026        MJDI ABDALNASER        RYIADH P.O BOX 3333

   10030        MANSOOR ABDO           RYIADH P.O BOX 4103

   10089        33333333333333333333
                         *                      *
```

نتائج البرنامج رقم ٢ الفصل الخامس صفحة ١٩٨

```
                                   S T O C K   L I S T
DATE : 84/11/24

   ITEM NO.      ITEM DESCRIPTION           QUANTITY      ITEM CODE
      10012       TABLES                        20             30
      10034       TELEPHONES                    23             40
      10044       CLASSES                      250             60
      10054       PENCILS                      125             50
      10055       NOTE BOOKS                   300            140
      10056       GLUE                         300            100
      10060       BOOKS                        200             80
      10064       CARTOON FILES                163            120
      10070       PENTEL PEN                   220            130
      10088       PLASTIC FILES                130            110
      10104       ARM CHAIRS                    23             10
```

نتائج برنامج رقم ٢ الفصل السابع صفحة ٢٤٩

DRAFT AMOUNT	DEPOSIT AMOUNT	NEW BALANCE
432,34	4530,00	+27497,66
500,00	3000,00	+35500,00
700,00	200,00	+4060,00
3000,00	3000,00	+6300,00
7000,00	2000,00	-700,00
5000,00	10000,00	+10550,00
00,00	2000,00	+22000,00
2000,00	3000,00	+7500,00
333,33	3300000,00	+3300000,00
800000,00	00,00	-793400,00
500000,00	4000,00	-495600,00
00,00	3000,00	+8500,00
300000,00	500000,00	+200000,00

شركة الكهرباء

قائمة بالسجلات المرفوضة من ملف الاستهلاك

صفحة رقم: 1

تاريخ القراءة	ملاحظات
1405/02/02	COUNTER OUT
1405/02/02	COUNTER OUT
1405/02/02	COUNTER OUT
1405/02/02	COUNTER OUT
1405/02/02	COUNTER OUT
1405/02/02	COUNTER OUT

ACCOUNT NO	ACCOUNT NAME	LAST BALANCE
20045	ZUHAIR AL KHAYAT	23400,00
20046	FATHI IMAM AL KHAIR	33000,00
20047	FIRAS AHMED JNAID	4560,00
20048	LINAH AL KHAYAT	6300,00
20049	KHALDOON AHMED JUNAID	4300,00
20050	HUDA IBRAHIM ALCHATTA	5550,00
20051	SALMA ZUHAIR ALKHAYAT	20000,00
20053	JAWHARA AL MOJEL	6500,00
20056	IBRAHIM AL SANEH	333,33
20057	NASER AL OHEBI	6600,00
20058	MAHMOOD MOHAMED ALI	400,00
20060	MOHAMED ABO SALEM	5500,00
20061	NAJEEB BEN SAEID	00,00

TOTAL OF ACTIVE RECORDS :	13
TOTAL OF LAST BALANCE :	116443,33
TOTAL OF DRAFT AMOUNTS :	1618965,67
TOTAL OF DEPOSIT AMOUNTS:	3834730,00
TOTAL OF NEW BALANCES :	+2332207,66

نتائج برنامج التمارين العامة تمرين ٢ صفحة ٤٨٦

21/11/1984: تاريــــخ

رقم المشترك	مقدار القراءه
10500	1550
10519	1550
10529	1550
10539	1550
10549	1550
10559	1550

نتائج برنامج التمارين العامة تمرين رقم ٥ صفحة ٤٩٢

8	9	10	11	12	13	14

PAGE : ZZ9

SOCIAL SECURITY SOCIAL FUNDS NET PAY SIGNATURE

Z999.99 ZZ99.99 ZZZ99.99

8	9	10	11	12	13	14

(1) Fold under

```
                                                  BRIGHT STAR COMPANY

DATE : 99/99/9999                                     PAYROLL LIST


   EMPLOYEE NO.   EMPLOYEE NAME           BASE SALARY     TRANSPORTATION

        ZZZ9         XXXXXXXXXXXXXXXXXXXX    ZZ999.99             ZZ9.99
```

```
NY
IBER RECORD                                                    PAGE :ZZ9

CRIPTION DATE       SUB. CODE          SUB. AREA  COUNTER  LAST READ
-  99  - 9999           99                 99      9999999   9999999
   *       *            *                  *          *         *
```

(1) Fold under

ELECTRIC COMPA

DATE : 99/99/9999 LIST OF INVALID SUBSCR

SUBSCRIBER NO. SUBS. NAME SUBS. ADDRESS SUBS

ZZ999 XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX 99

8 9 10 11 12 13 14

ST

T. ALLOWANCES PENSION NET PAY

ZZ99.99 ZZ99.99 ZZZ99.99

8 9 10 11 12 13 14

(1) Fold under

PAYROLL LI

EMPLOYEE NO. EMPLOYEE NAME SALARY TRANSPOR

ZZ999 XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX ZZZ99.99 ZZ9.99

BALANCE COMMENTS

ZZZZZZ9.99 XXXXXXXX

(1) Fold under

CUSTOMER REPORT

CUSTOMER No. CUSTOMER NAME

ZZZZ99 XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

8 9 10 11 12 13 14

PAGE NO.: ZZ9

TEM PRICE MINIMUM LEVEL COMMENTS

ZZ9.99 ZZZ99 XXXXXXXX

8 9 10 11 12 13 14

(1) Fold under

```
                                                            STOCK LIST
DATE  : 99/99/1999

    ITEM NO.    ITEM DESCRIPTION         QUANTITY      ITEM CODE

    ZZZ99     XXXXXXXXXXXXXXXXXXXX         ZZZZ9          99
```

ANY
RECORDS   PAGE :ZZ9

SUBSCRIPTION DATE   SUB.CODE   SUB.AREA   COUNTER NO.

99-99-9999   99   99   9999999

(1) Fold under

ELECTRIC COMP

DATE : 99/99/1999 LIST OF SUBSCRIBER

SUBSCRIBER NO. SUB. NAME SUB. ADDRESS

99999 XXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXX

قائمة بالم

تاريخ : 9999-99-99

رقم المشترك	اسم المشترك	عنوان المشترك
9ZZZZ	XXXXXXXXXXXX XXX	XXXXXXXXXXXXXXXXXXXXXXXXX XXX

عدد المشتر

مقدار الدخ

مجموع المب

⊲ (1) Fold under

شركة الكهرباء

مشتركين واستهلاك الشهري

صفحة رقم : ZZ9

رمز المنطقة	رمز الاشتراك	رقم العداد	مقدار الاستهلاك	المبلغ المستحق
99	99	9999999	9999999	ZZ799,99

الخلاصة النهائية

المشتركين : ZZZ999

الاموال المدفوعة : ZZZZZ9999

المبالغ المستحقة : ZZZZ9999,99

8 9 10 11 12 13 14

NK

LANCES PAGE : ZZ9

POSIT AMOUNT DRAFT AMOUNT NEW BALANCE

99999.99 99999.99 9999999.99

RDS : 9999

ES : 99999999.99

UNTS : 99999999.99

UNTS : 99999999.99

CES : 99999999.99

8 9 10 11 12 13 14

(1) Fold under

COMMERCIAL BA

DATE : 99/99/1999 REPORT FOR NEW BA

ACCOUNT NO. ACCOUNT NAME LAST BALANCE DR

99999 XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX 9999999.99

TOTAL OF ACTIVE RECO

TOTAL OF LAST BALANC

TOTAL OF DEPOSIT AMO

TOTAL OF DRAFT AMO

TOTAL OF NEW BALAN

رقم السودان

٤٤٩٩٩

(1) Fold under

شركة البنين

ملاحظات	خارج الدوام	المرتب	اسم الموظف
XXXXXXXXXX	ZZ99,99	ZZZ99,99	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

مجموع الاجور : ZZZ999,99

مجموع خارج الدوام : ZZ999,99

مجموع عدد العمال : Z999

تاريخ : 99/99/9999

رقم المشترك مقدار القراءة

ZZ999 9999999

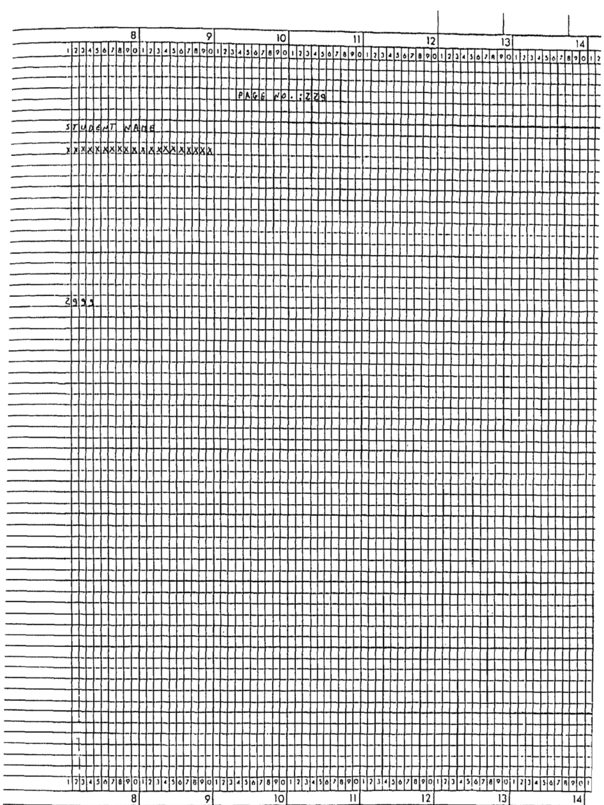
(1) Fold under

شركة الكهرباء

قائمة التحويلات المرفوضة من ملف الاستهلاك مطبوعة رقم د ٢٩ ج

تاريخ القراءة ملاحظات

وو - وو - وووو XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX



(1) Fold under

```
                                                 KING SAUD UNIVERSITY

DATE : 99/99/9999                                  STUDENTS REPORT


              COLLAGE CODE            YEAR NO.        STUDENT NO.

                  99                    99              ZZ999              XXXXXXXXXX



    TOTAL OF YEAR NO. 99 IS  :      ZZ99

**TOTAL OF COLLAGE CODE 99 IS :ZZ999







                                                 TOTAL OF ALL STUDENTS ARE : ZZ
```

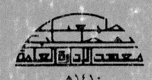
الإدارة العامة
معهد
طبعت بمطابع

١٤٠٥هـ

جدول بالتصويبات الخاصة بكتاب البرمجة التطبيقية بلغة الكوبول

رقم الصفحة	رقم السطر	العبارة الخطأ	العبارة الصواب
٢٥	١٢	ـ قسم البيانات	٣ ـ قسم البيانات
٩٥	٧	أكبر من ٣٦	أقل من ٣٦
١٠٤	٢١	CONTAINS BO	CONTAINS 80
١٤٩	٧,٦	]	],....
١٥٧	١	أجر العام	أجر العامل
١٦٣	٥	217.	217
١٧٥	٨	1234.	12345.
١٧٦	١	مثال :	مثال : (١)
١٧٦	يضاف لنهاية الصفحة	(١) قد تختلف الصيغة فى بعض الأجهزة.	
١٨٦	٤	OVERLOW	OVERFLOW
٢٠٨	٥	- STORAG	- STORAGE
٢٣٥	١٢	رقم الطالب تصاعدياً	اسم الطالب تصاعدياً
٢٤٦	٢	- 4]	- 4],...
٢٧٨	١٠	READ MSTR	READ TRNS
٤٠٥	٨	وسائل تحذير	رسائل تحذير
٤١٨	١٠	COMMAND > INPUT	COMMAND INPUT >
٤٦٢	الشكل أسفل الصفحة	عمليات الحذف	عمليات التعديل
٤٦٢	الشكل أسفل الصفحة	عمليات تعديل	عمليات حذف
٥١٣	٦	تحذف الأسطر اعتباراً من السطر ٦ لنهاية الصفحة لتكرارها	
٥٢٤	١	النقطة (الفاصلة العشرية)	. النقطة (الفاصلة العشرية)
٥٢٤	٢	الخط المائل	/ الخط المائل

السعر ٣٤ ريالاً



١٤١٠هـ